Programme du cours

- Introduction et contexte
- Réseaux de neurones et optimisation
- Deep Computer Vision
- Deep Sequence model
- Deep Generative model (vision)

Génération d’images à partir d’un prompt

A Person Transported To Another World Through A Wormhole

The Irony of IronMan Ironing with the Iron in his Iron Suit

A Man looking At The Starry Sky By Vincent Van Gogh

Source : Reddit StableDiffusion, https://stability.ai/blog/stable-diffusion-announcement

Génération de visages

Source : StyleGAN. Karras et al, 2019.

Génération de visages - Deepfake

Source : StyleGAN. Karras et al, 2019.

Génération de visages - Deepfake

Génération de visages - Deepfake

Génération de vidéos

Génération de vidéos

Génération de vidéos

Discriminative vs Générative

Discriminative vs Générative

Modèle Discriminative

Donnée : (x, y)

Avec x les données et y les labels

Objectif :

Apprentissage de la distribution de probabilité

Tâches :

Classification et régression

Discriminative vs Générative

Modèle Discriminative

Donnée : (x, y)
Avec x les données et y les labels

Objectif :
Apprentissage de la distribution de probabilité

Tâches :
Classification et régression

Modèle Génératif

Donnée : x
Avec x les données et pas de labels

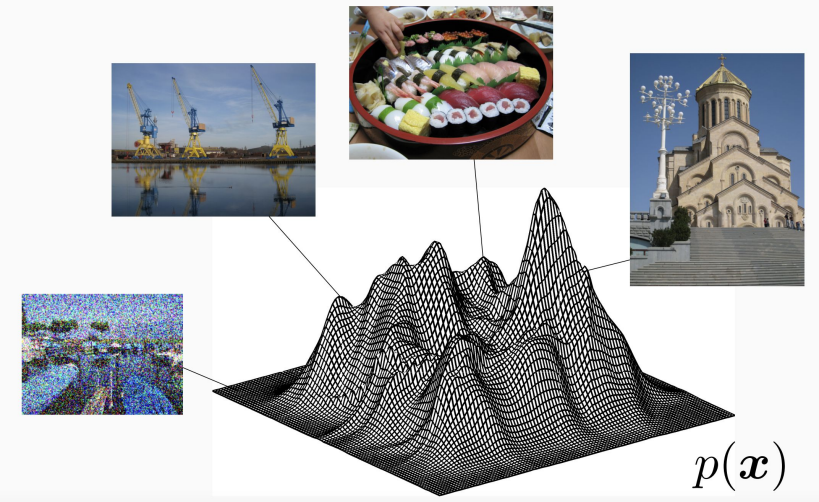
Objectif :
Apprentissage de la distribution de probabilité

Dans le but de générer de nouveaux échantillons de données similaires à l'ensemble de données

Applications

- **Génération d'images** : génération d'art, création de deep fake, de décors pour jeux vidéos, génération d'images à partir de descriptions textuel, … etc.
- **Génération de textes** : chatbots, génération de codes, génération de textes à partir d'images, … etc.
- **Détection d'anomalie** : modélisation de la distribution d'une source de données. Lorsque de nouvelles données s'écartent significativement de cette distribution, elles peuvent être signalées comme une anomalie.
- **Découverte de médicaments** : Dans le domaine pharmaceutique, les modèles génératifs peuvent être utilisés pour découvrir de nouvelles molécules ou optimiser les molécules existantes pour le développement de médicaments.
- **Outlier detection:** voiture autonome → detecte quand un évènement anormal survient
- Augmentation de données
- Génération de voix

Génération d'images



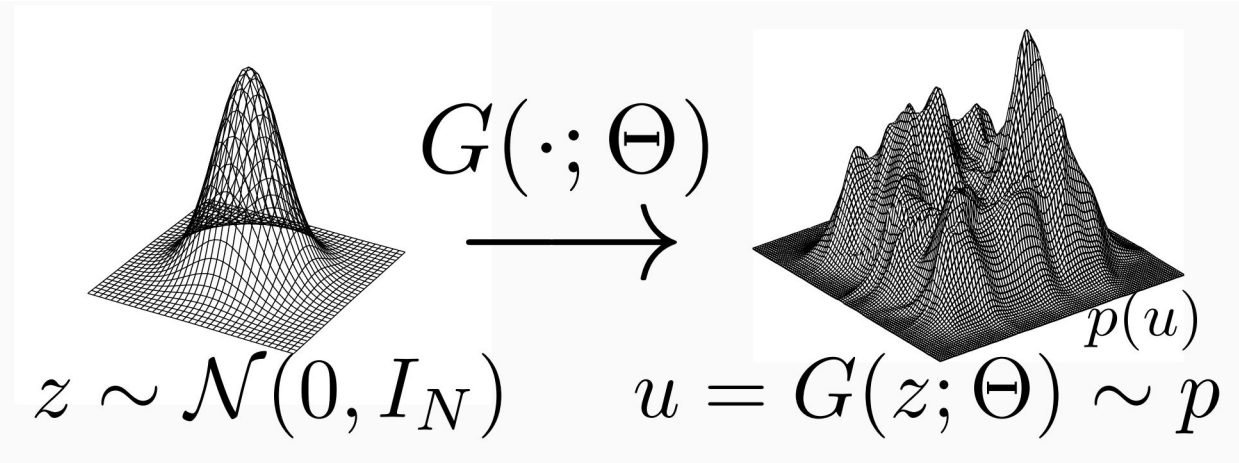
1) Apprentissage de la distribution de probabilité

2) Génération d'échantillons à partir de cette distribution

Source : Charles Deledalle

Génération d'images

1) Apprentissage de la distribution de probabilité
2) Génération d'échantillons à partir de cette distribution



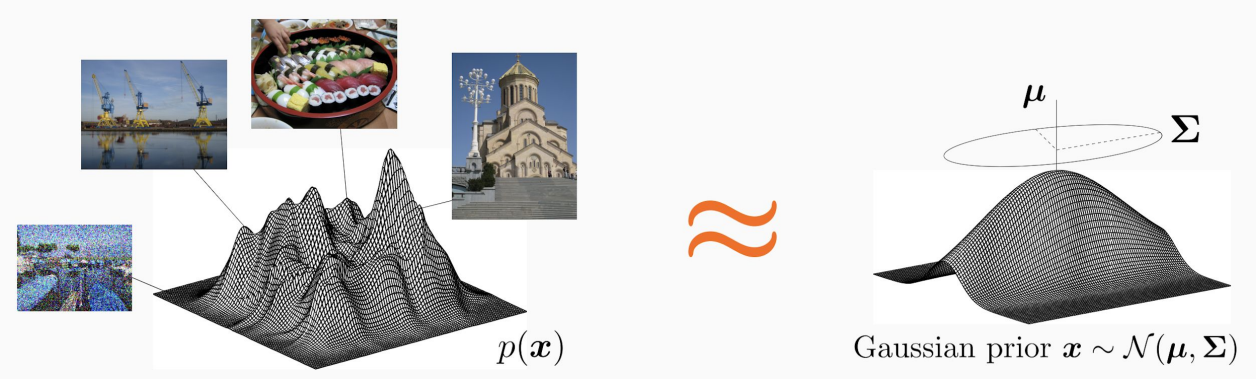
- z est appelé variable latente
- Le modèle G (souvent appelé générateur) peut être un réseau de neurones !

Modèle Gaussien

Considérons un modèle Gaussien pour la distribution des images avec pixels :

HYPOTHESE: la distribution de données est gaussienne (ou mixture de gaussienne)

- μ : image moyenne
- $\sum$: matrice de covariance associée aux images.



Source : Charles Deledalle

Modèle Gaussien

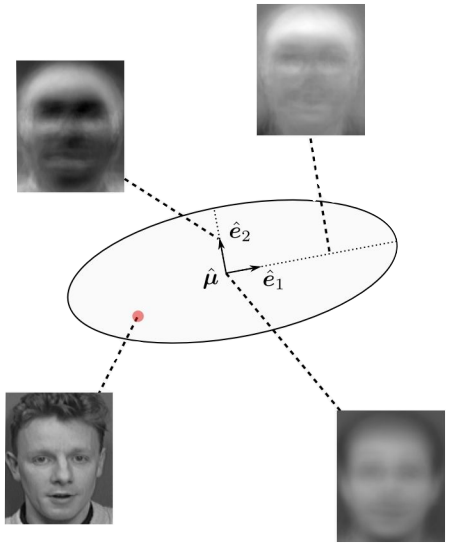
- Soit un ensemble de données d'apprentissage d'images :

- Estimation de la moyenne

- Estimation de la matrice de covariance

Source : Charles Deledalle

Modèle Gaussien



Source : Charles Deledalle

Modèle Gaussien

Générer un échantillon de la distribution gaussienne

Source : Charles Deledalle

Modèle Gaussien

Générer un échantillon de la distribution gaussienne

Source : Charles Deledalle

Modèle Gaussien

Générer un échantillon de la distribution gaussienne

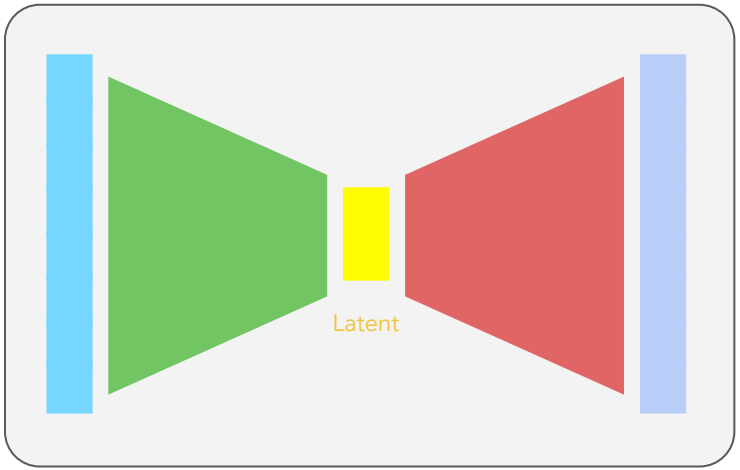
- Le modèle ne permet pas la génération de visage réaliste.
- **L'hypothèse d'une distribution gaussienne est trop simpliste.**
- Chaque image générée n'est qu'une combinaison linéaire aléatoire des vecteurs propres.
- Le générateur correspond à un réseau neuronal linéaire (sans non-linéarités).

Source : Charles Deledalle

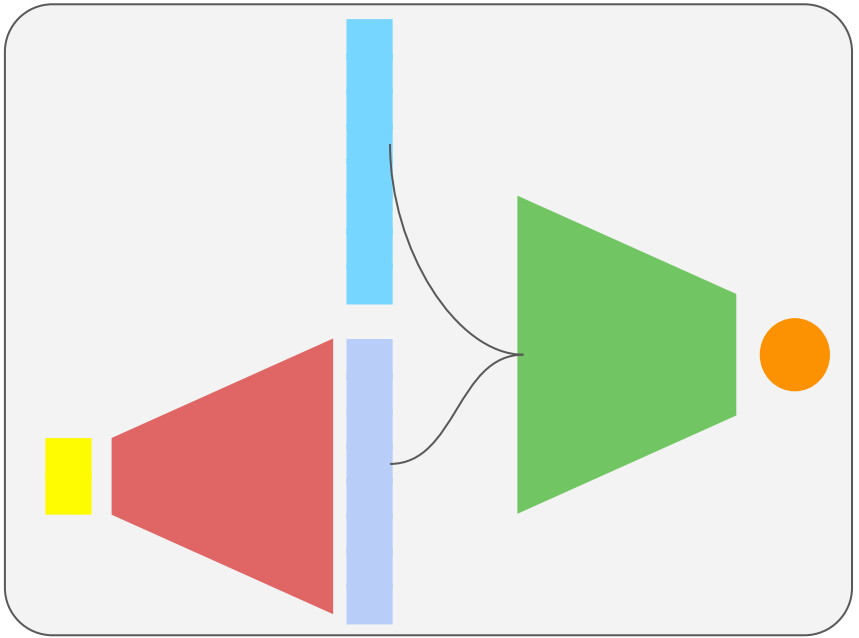
Solution: apprendre la distribution

Solution: apprendre la distribution

Modèles à variable latente



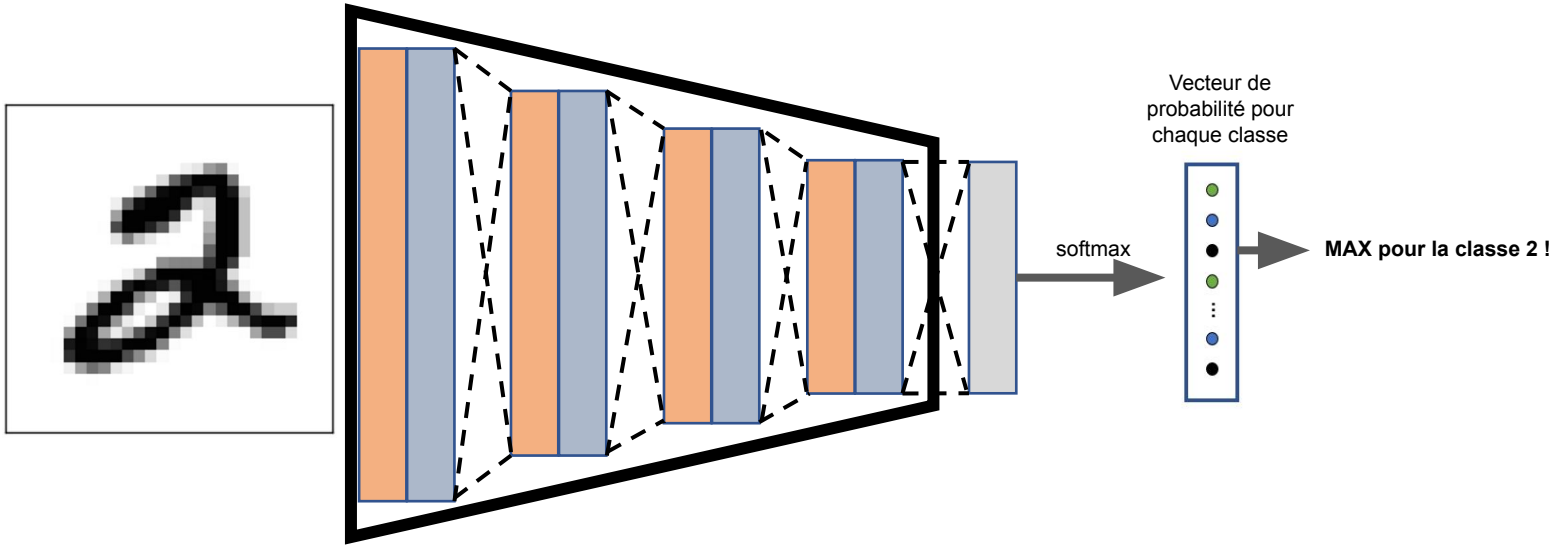
(Variational) Autoencoder



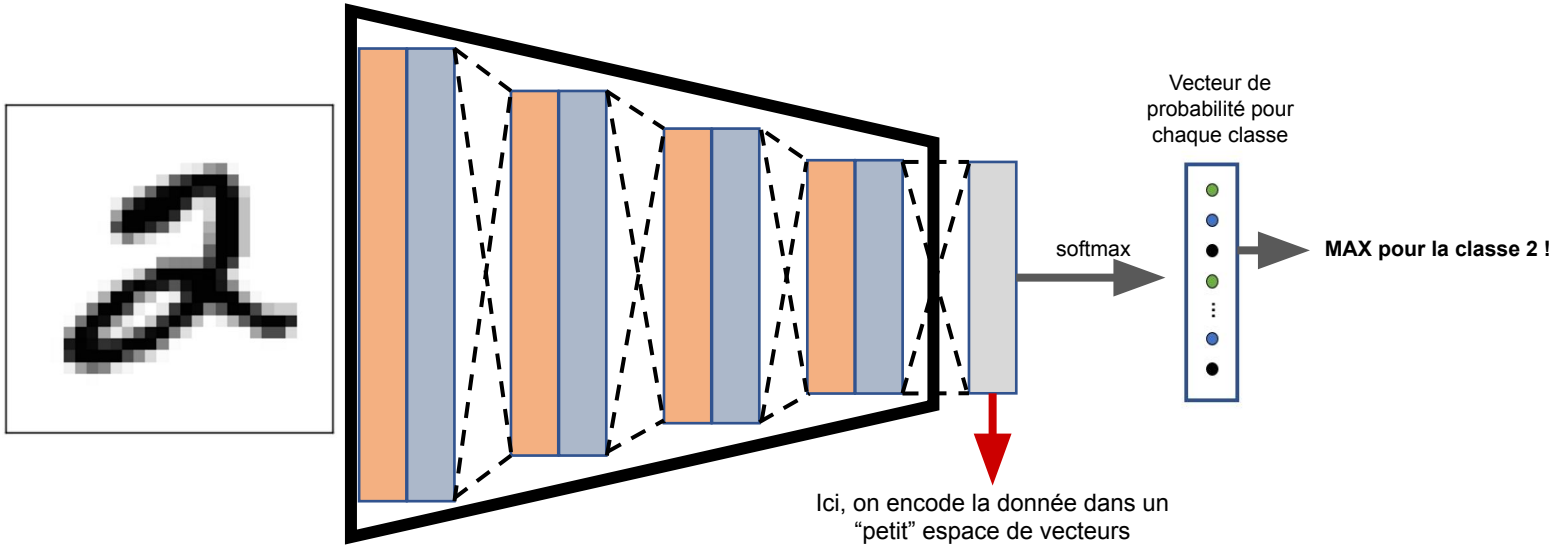
Generative Adversarial Networks

Auto-encoder

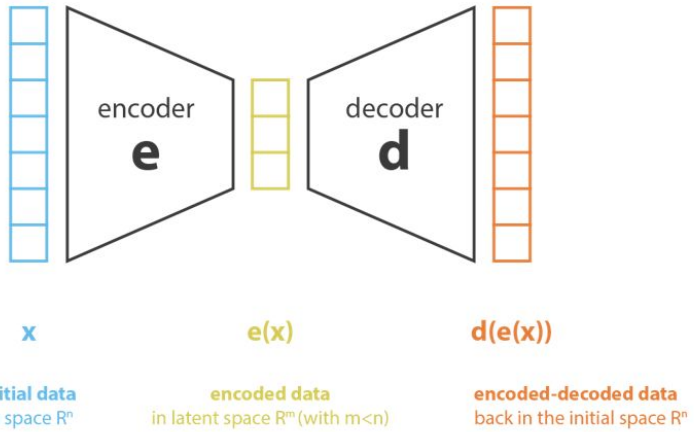
Auto-encoder



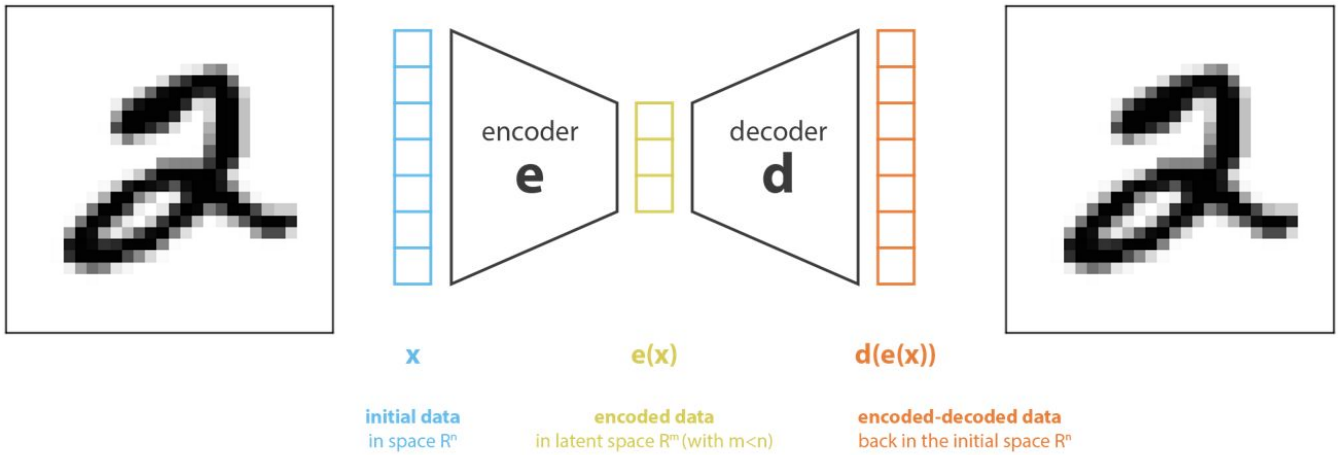
Auto-encoder



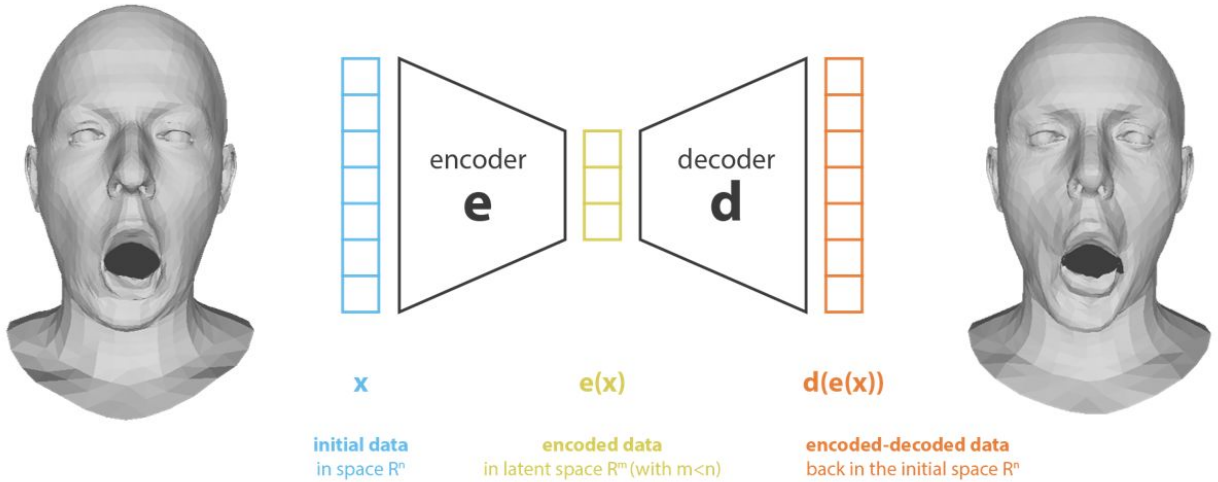
Auto-encoder



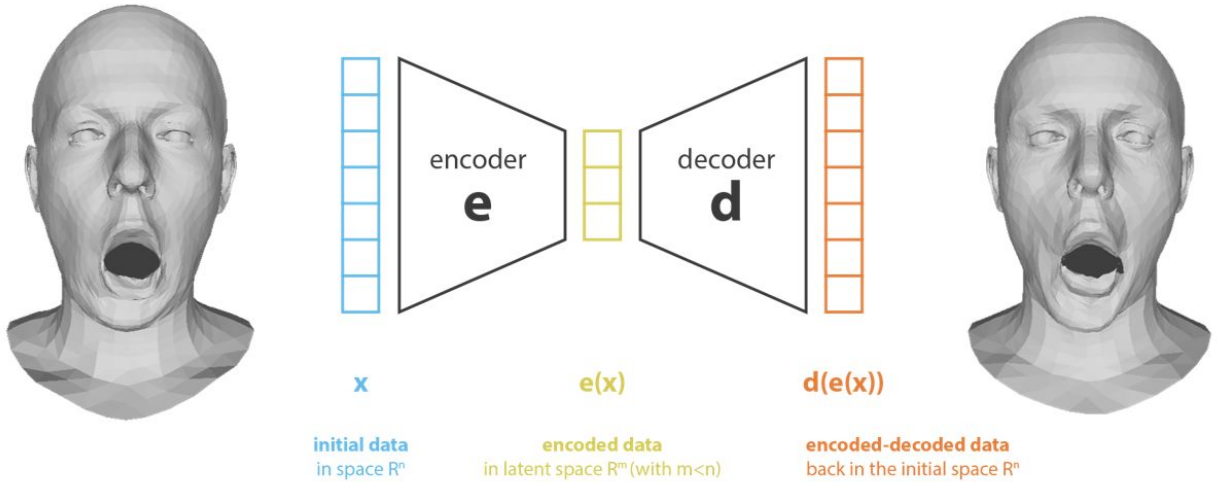
Auto-encoder



Auto-encoder



Auto-encoder



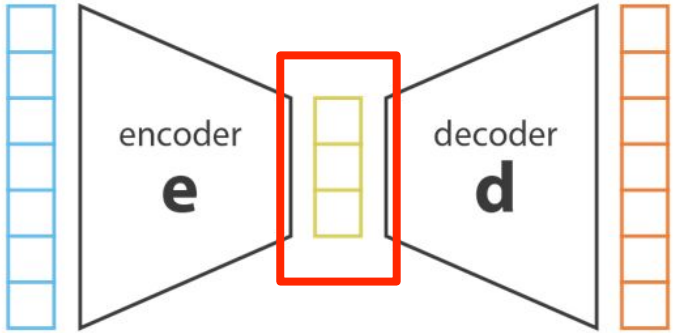
$$loss_{AE} = \|x - \hat{x}\|_1 = \frac{1}{N}\sum_i |x_i - \hat{x}_i|$$

Espace latent

- L'**espace latent** fait référence à un espace multidimensionnel euclidien abstrait contenant des valeurs de caractéristiques que nous ne pouvons pas interpréter directement, mais qui encodent une représentation interne significative d'événements observés.

- Les variables latentes ne sont pas directement mesurables ou observables. Elles sont plutôt déduites ou estimées sur la base des données observées.

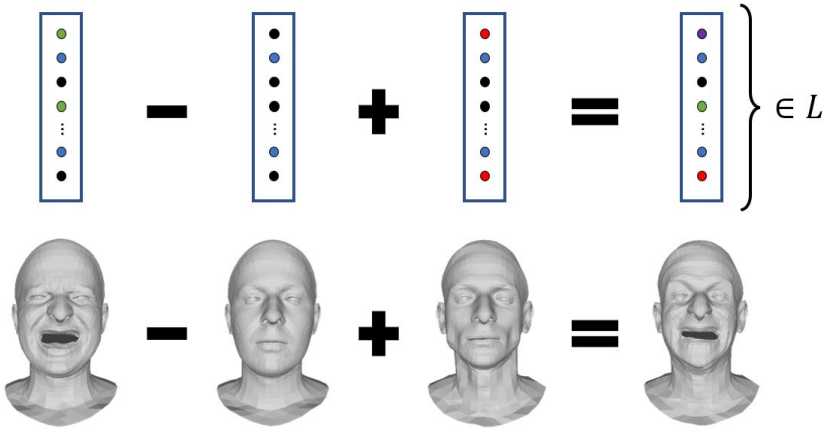
- Pour que ça marche, on veut forcer la dimension de l'espace latent à être petite: cela force le modèle à apprendre une compression "intelligente" de la donnée.



Espace latent

- L'**espace latent** fait référence à un espace multidimensionnel euclidien abstrait contenant des valeurs de caractéristiques que nous ne pouvons pas interpréter directement, mais qui encodent une représentation interne significative d'événements observés.

- Les variables latentes ne sont pas directement mesurables ou observables. Elles sont plutôt déduites ou estimées sur la base des données observées.



Autoencoders

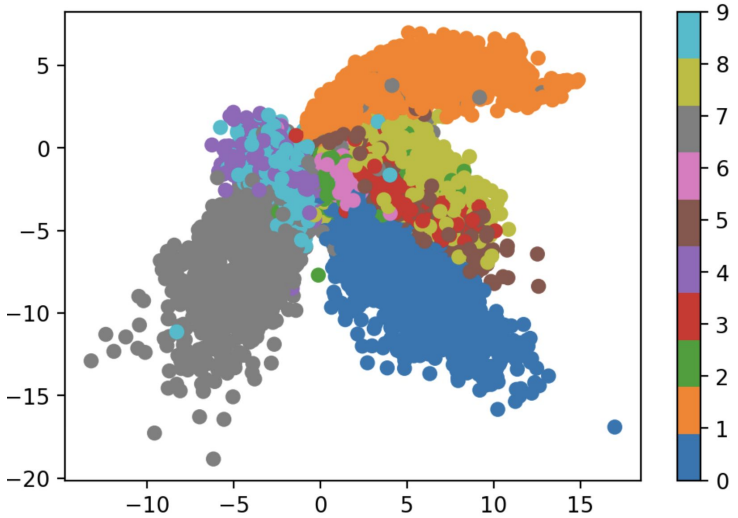
La dimension de l'espace latent pour les exemples ci-dessous est égale à 2

Image jeu de données test :

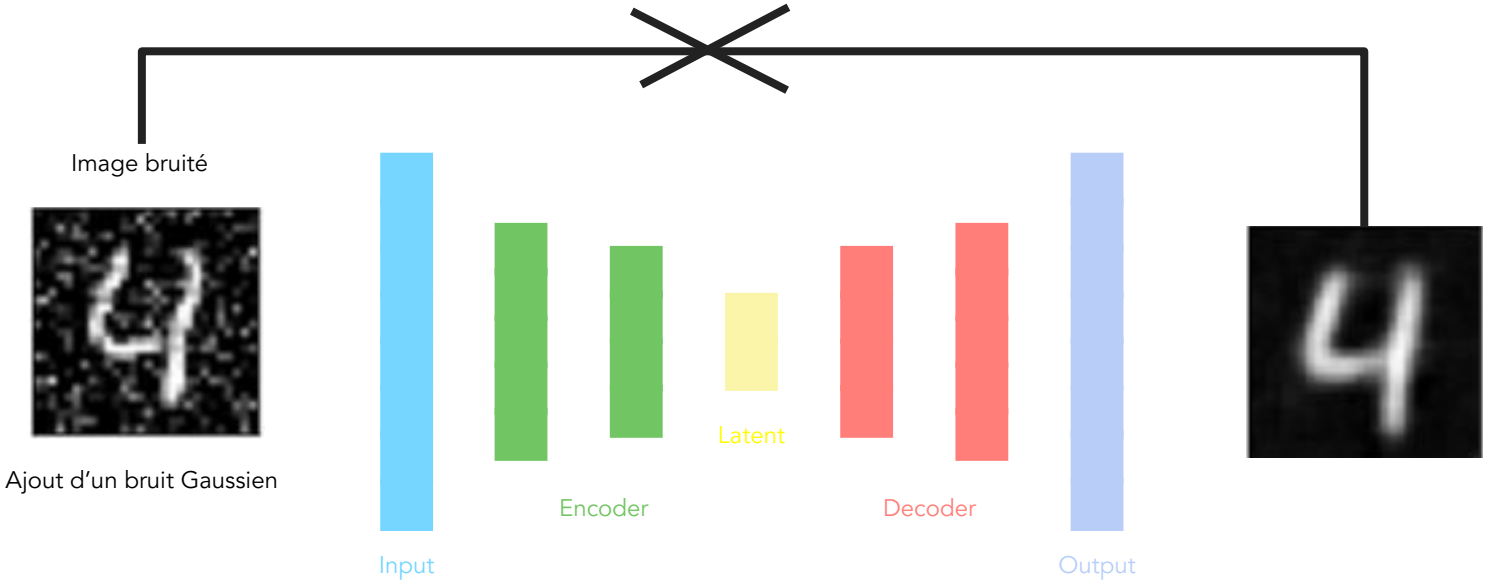
Reconstruction

Autoencoders

Représentation pour un ensemble de données test de leurs espace latents associés



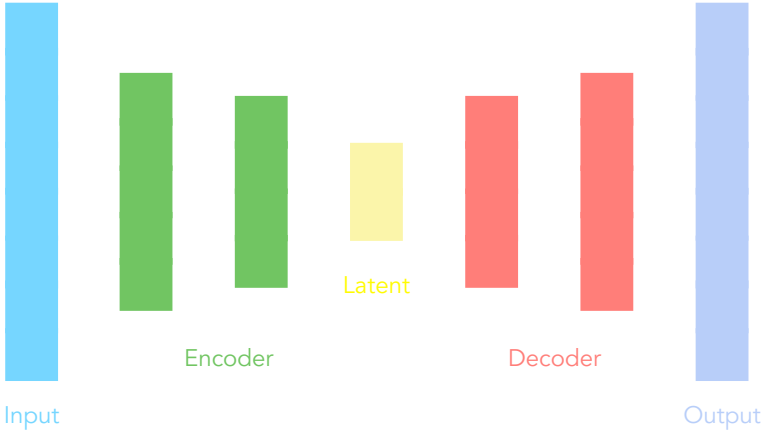
Denoising Autoencoders

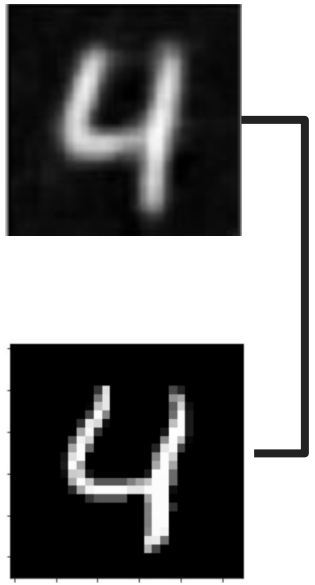


Denoising Autoencoders

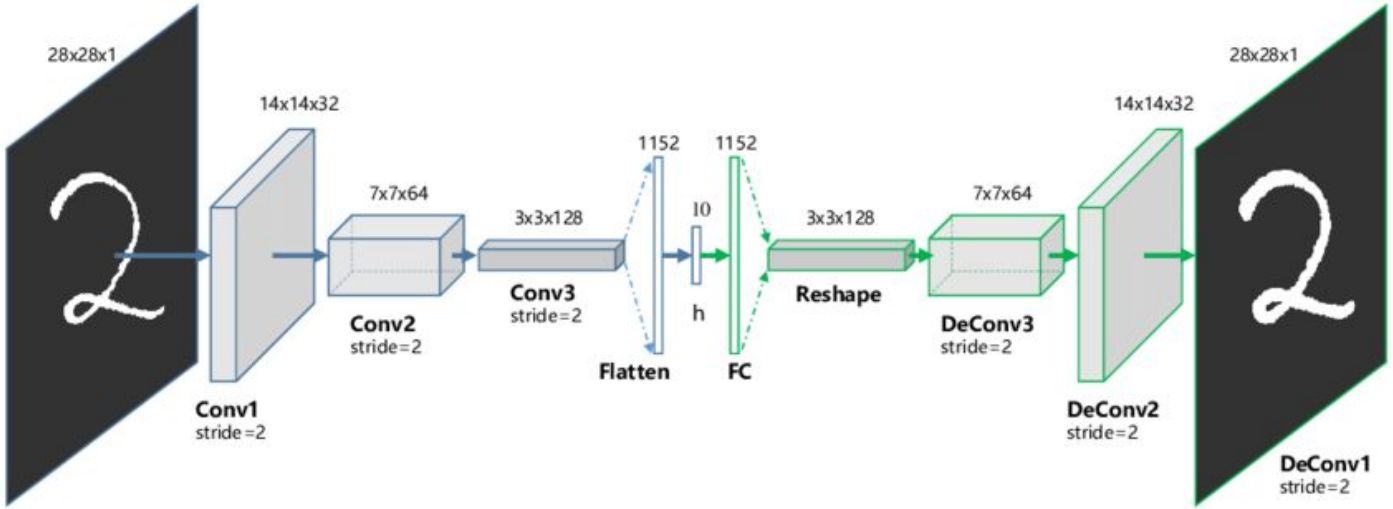
Image bruité

Ajout d'un bruit Gaussien





Convolutional Autoencoders



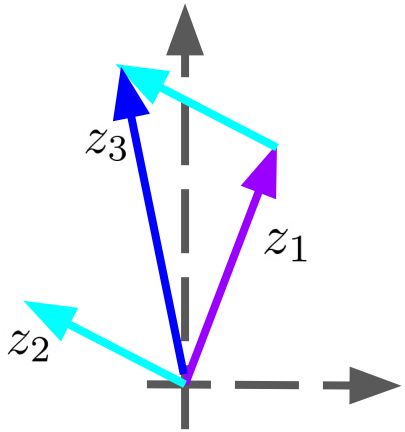
Guo et al. Deep Clustering with Convolutional Autoencoders, 2017.

Autoencoders - espace latent

L'espace latent est un espace dont la structure est simple *à priori* = compact, euclidien, …

→ Opération arithmétiques (+, -) possible

$$z_1 + z_2 = z_3$$

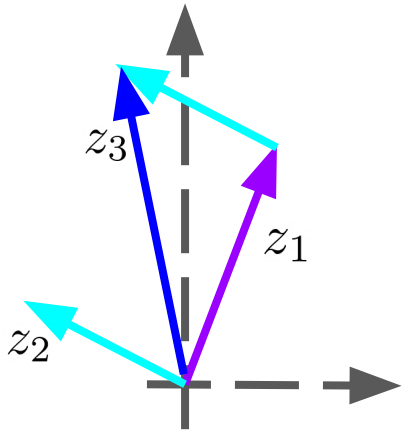


Autoencoders - espace latent

L'espace latent est un espace dont la structure est simple *à priori* = compact, euclidien, …

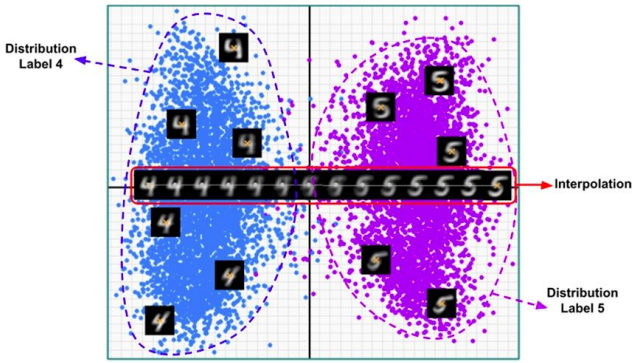
→ Opération arithmétiques (+, -) possible

$$z_1 + z_2 = z_3$$



→ Interpolation linéaire entre vecteurs latent

$$z_t = (1-t).z_1 + t.z_2$$



Autoencoders - limitations

Peut-on utiliser l'espace latent pour générer de nouvelles données ?

Autoencoders - limitations

Peut-on utiliser l'espace latent pour générer de nouvelles données ?

- Les Autoencoder ont pour objectif l'apprentissage d'une représentation compact, riche à partir de données. Par conséquent, les autoencoders ne se concentrent pas sur l'apprentissage de la distribution sous-jacente des données, qui est une exigence clé pour la génération de données.

Autoencoders - limitations

Peut-on utiliser l'espace latent pour générer de nouvelles données ?

- Les Autoencoder ont pour objectif l'apprentissage d'une représentation compact, riche à partir de données. Par conséquent, les autoencoders ne se concentrent pas sur l'apprentissage de la distribution sous-jacente des données, qui est une exigence clé pour la génération de données.

- Si nous décidons d'échantillonner un vecteur latent , il est fort possible que celui-ci n'est pas été vu durant l'entraînement et qu'en conséquence le modèle génère une donnée non cohérente.

Autoencoders - limitations

Peut-on utiliser l'espace latent pour générer de nouvelles données ?

- Les Autoencoder ont pour objectif l'apprentissage d'une représentation compact, riche à partir de données. Par conséquent, les autoencoders ne se concentrent pas sur l'apprentissage de la distribution sous-jacente des données, qui est une exigence clé pour la génération de données.

- Si nous décidons d'échantillonner un vecteur latent , il est fort possible que celui-ci n'est pas été vu durant l'entraînement et qu'en conséquence le modèle génère une donnée non cohérente.

- Absence de distribution de probabilité explicite. Cela leur permettraient de générer des points de données par échantillonnage à partir d'une distribution de probabilité bien définie.

Autoencoders - limitations

Peut-on utiliser l'espace latent pour générer de nouvelles données ?

- Les Autoencoder ont pour objectif l'apprentissage d'une représentation compact, riche à partir de données. Par conséquent, les autoencoders ne se concentrent pas sur l'apprentissage de la distribution sous-jacente des données, qui est une exigence clé pour la génération de données.

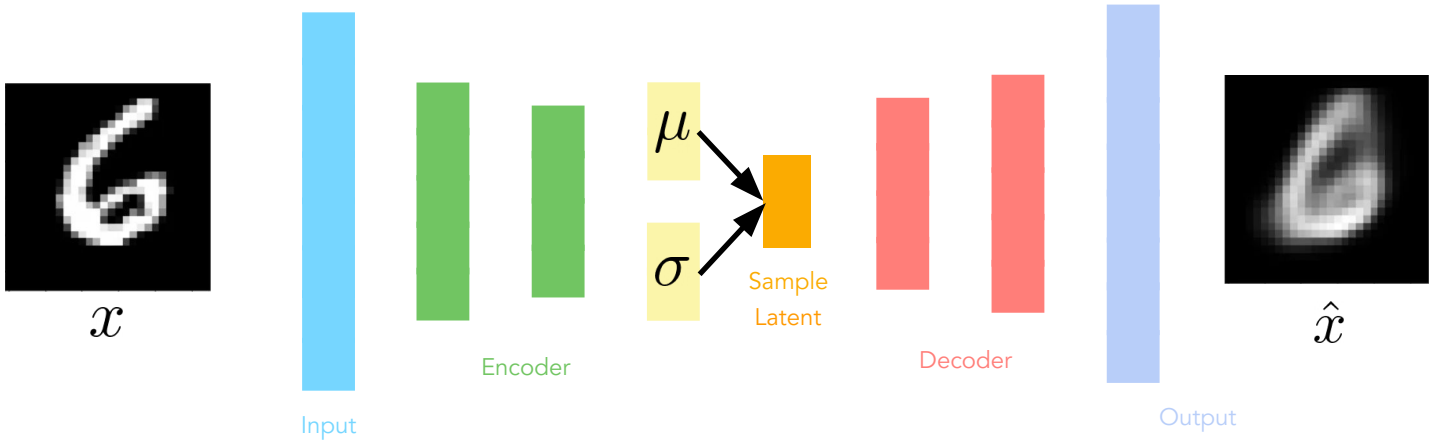
- Si nous décidons d'échantillonner un vecteur latent , il est fort possible que celui-ci n'est pas été vu durant l'entraînement et qu'en conséquence le modèle génère une donnée non cohérente.

- Absence de distribution de probabilité explicite. Cela leur permettraient de générer des points de données par échantillonnage à partir d'une distribution de probabilité bien définie.

- L'espace latent n'est pas continue.

Variational Autoencoders (VAEs)

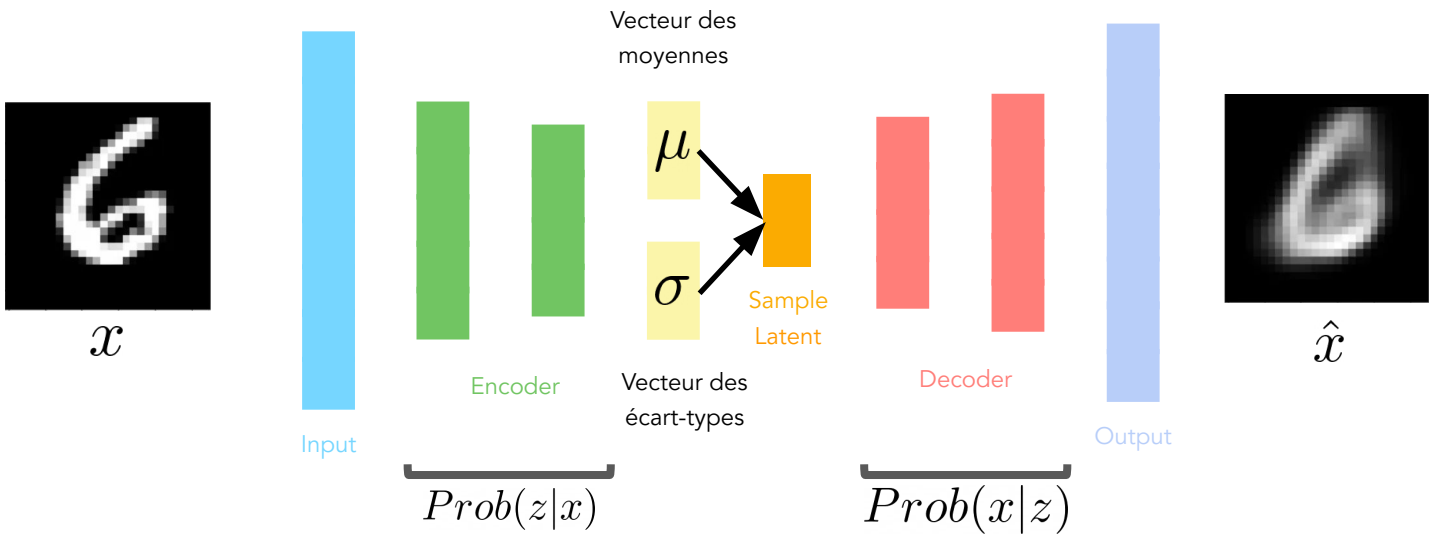
Apprentissage d'une correspondance probabiliste entre les données de haute dimension et un espace latent de dimension inférieur



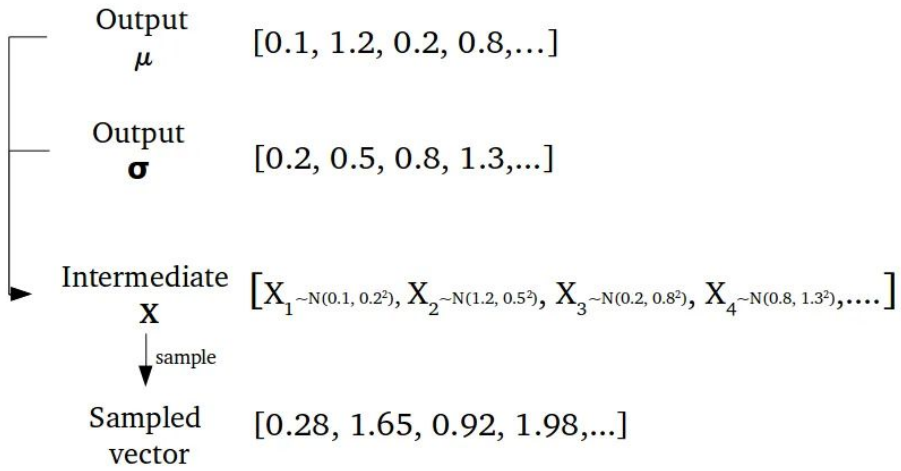
Les données d'entrée sont encodées sous forme de distribution dans l'espace latent

Variational Autoencoders (VAEs)

Apprentissage d'une correspondance probabiliste entre les données de haute dimension et un espace latent de dimension inférieur

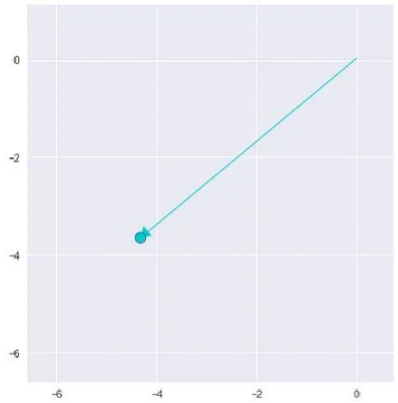


VAE - Espace latent

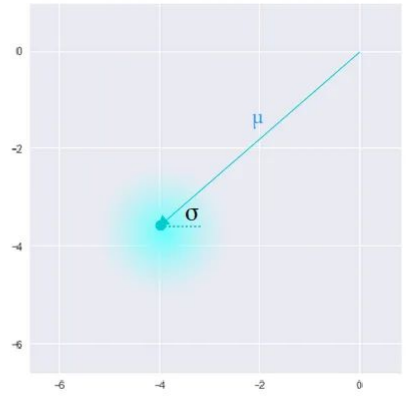


VAE - Espace latent

Le vecteur moyenne contrôle le point autour duquel l'encodage doit être centré, tandis que l'écart type contrôle la "zone autour"



Standard Autoencoder
(direct encoding coordinates)

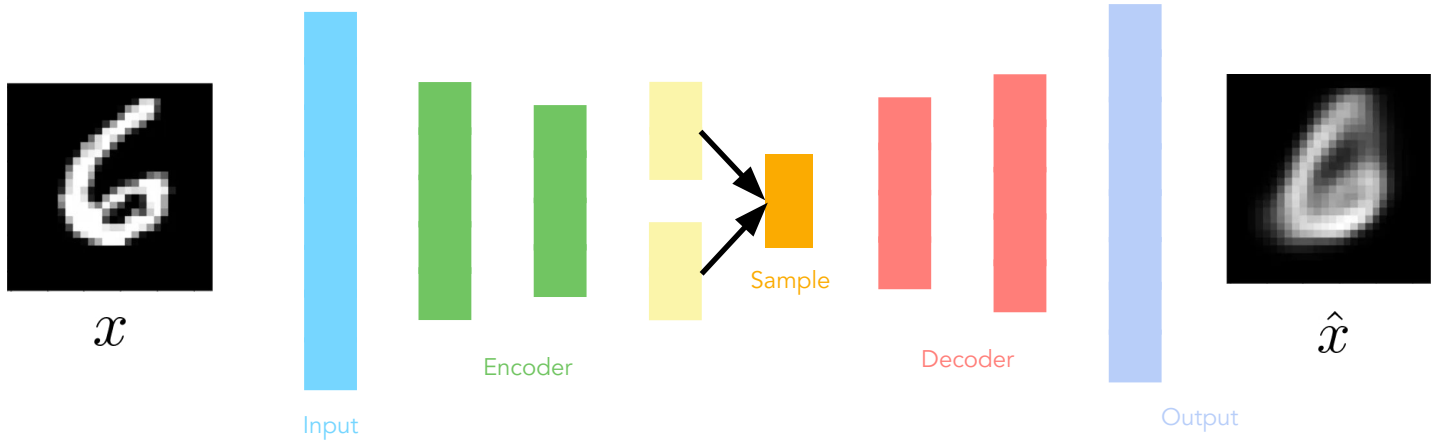


Variational Autoencoder
(μ and σ initialize a probability distribution)

VAE - Espace latent

- Étant donné que les codages sont générés aléatoirement à partir de n'importe quel point de la distribution, le décodeur apprend que plusieurs points proches dans l'espace latent sont tous associés à un échantillon de cette classe.
- Cela permet au décodeur de décoder à la fois des encodages uniques et des variations légères, car il est exposé à diverses versions de l’encodage pour la même entrée pendant l'entraînement.

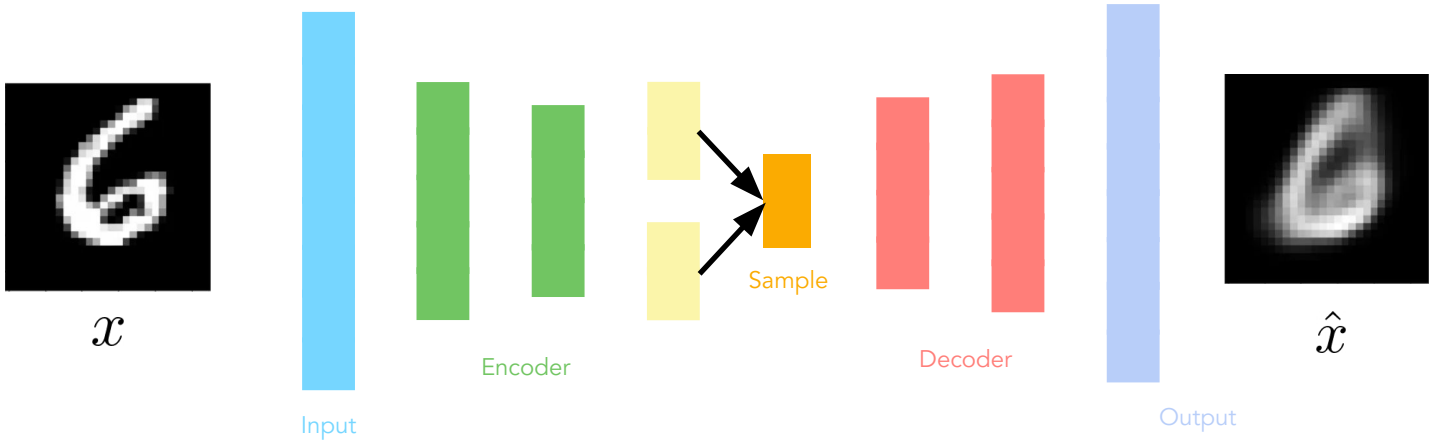
VAE - Optimisation



$$l(x, \hat{x}) = \|x - \hat{x}\|^2$$

loss reconstruction (MSE)

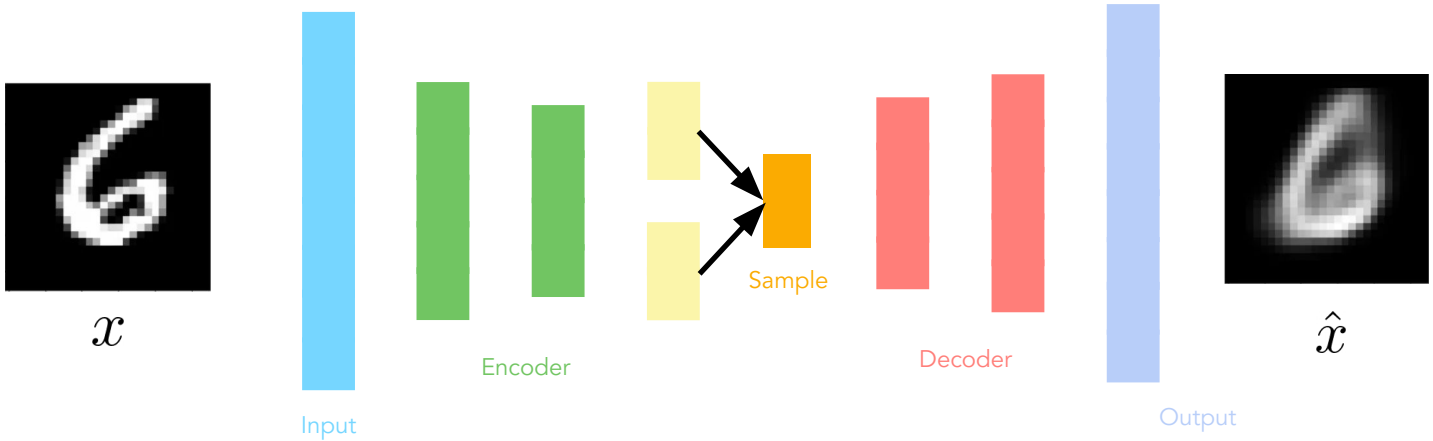
VAE - Optimisation



$$l(x, \hat{x}) = \|x - \hat{x}\|^2 + \lambda KL((\mu, \sigma), (0, I))$$

loss reconstruction (MSE) + terme de régularisation (divergence de Kullback-Leibler)

VAE - Optimisation

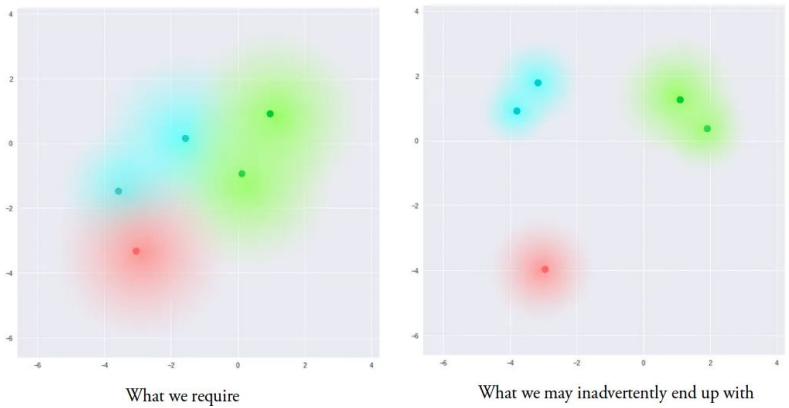


$$KL((\mu, \sigma), (0, I)) = \sum_i \sigma_i^2 + \mu^2 - \log(\sigma_i) - 1$$

loss reconstruction (MSE) + terme de régularisation (divergence de Kullback-Leibler)

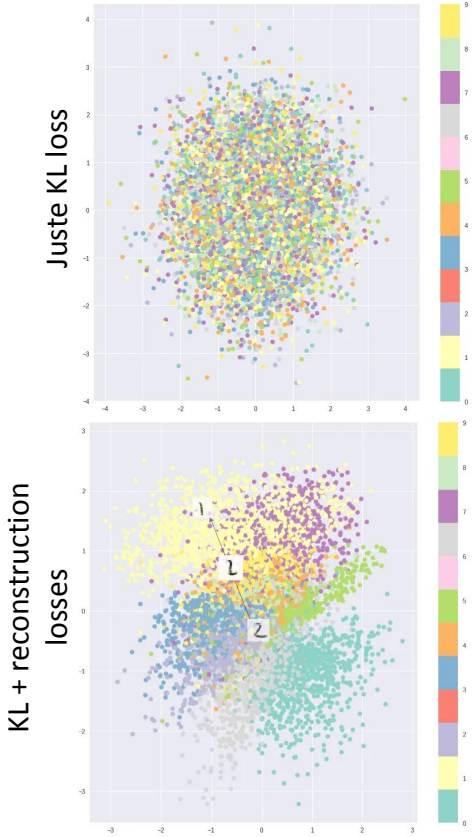
VAE - Régularisation

Nous désirons un chevauchement entre les échantillons non similaire, afin de faire l'interpolation entre les classes.



Cependant, il n'y a pas de limite sur les valeurs de moyenne et d'écart-type que peuvent prendre et le modèle peut apprendre des valeurs très distinctes pour séparer les classes (image de droite).

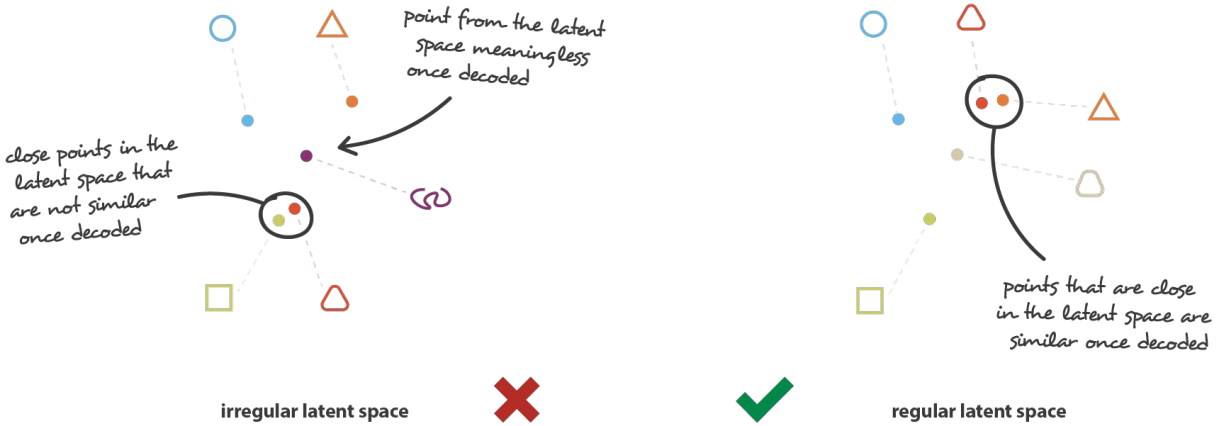
VAE - Régularisation



- Ce terme de régularisation encourage à distribuer tous les encodages de manière uniforme autour du centre de l'espace latent.
- Pénalise le réseau si il essaie de regrouper les échantillons dans des régions spécifiques.

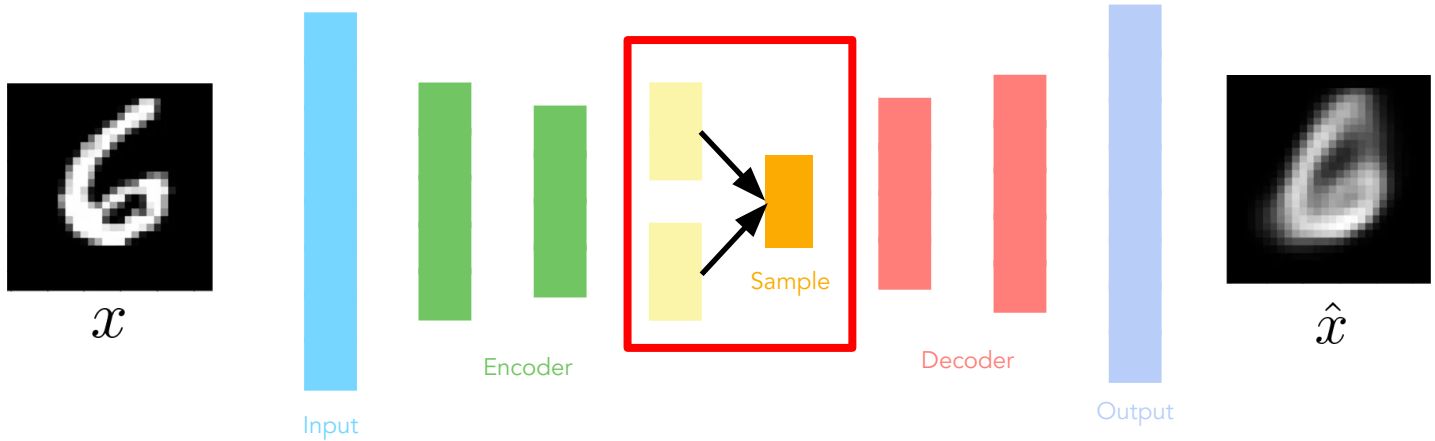
Intuition derrière le terme de régularisation

- **Continuité** : deux points proches dans l'espace latent ne doivent pas donner deux contenus complètement différents une fois décodés
- **Exhaustivité** : pour une distribution choisie, un point échantillonné dans l'espace latent doit donner un contenu "significatif" une fois décodé

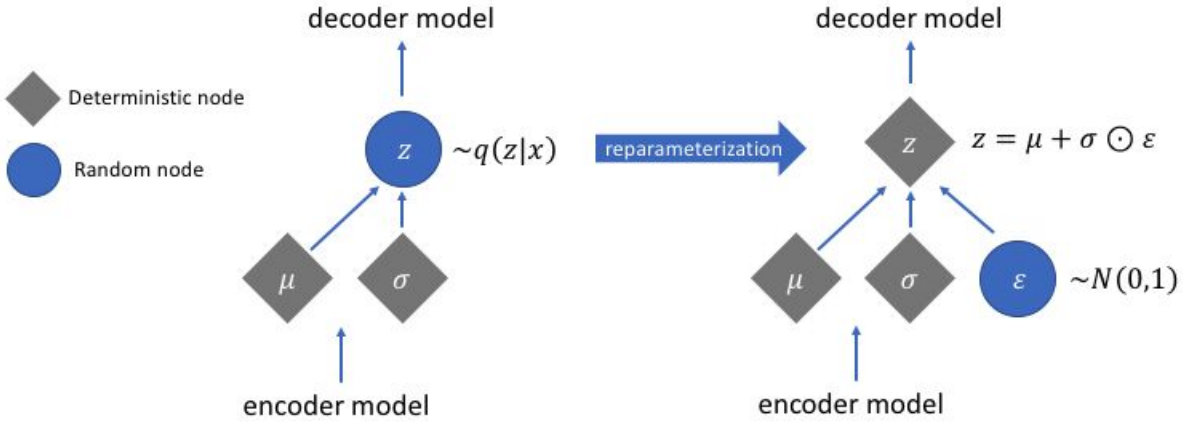


VAE - Computation graph

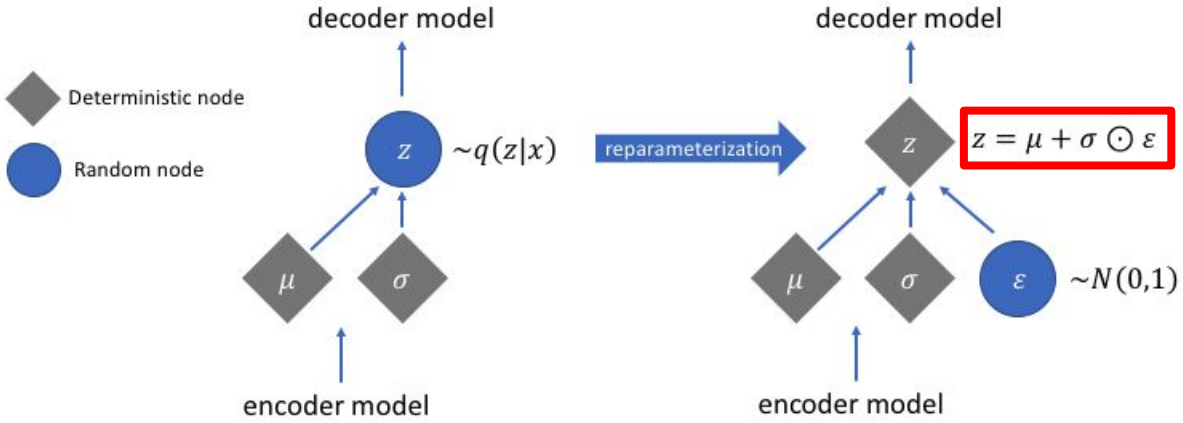
Il n'est pas possible de rétro propager le gradient à travers une couche d' échantillonnage ici !



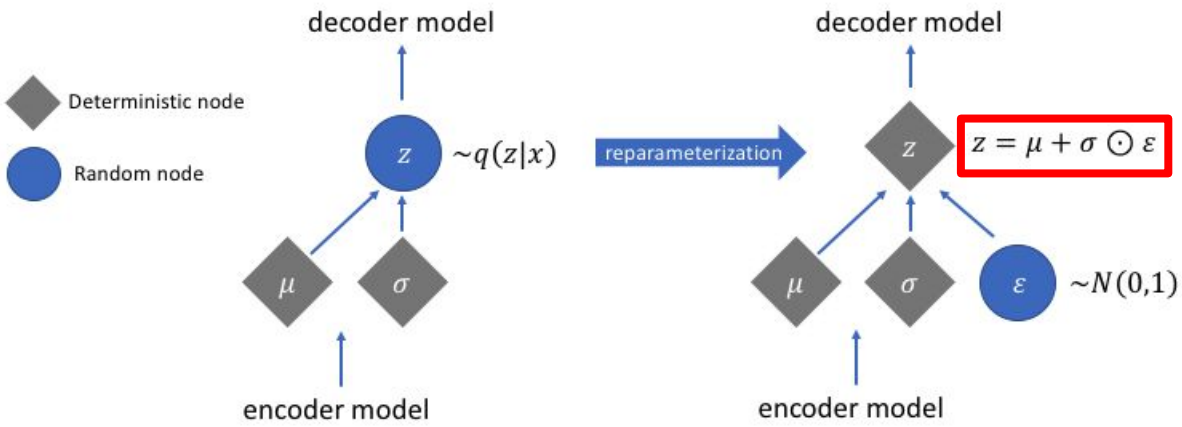
VAE - Reparamétrisation de la couche d'échantillonnage



VAE - Reparamétrisation de la couche d'échantillonnage



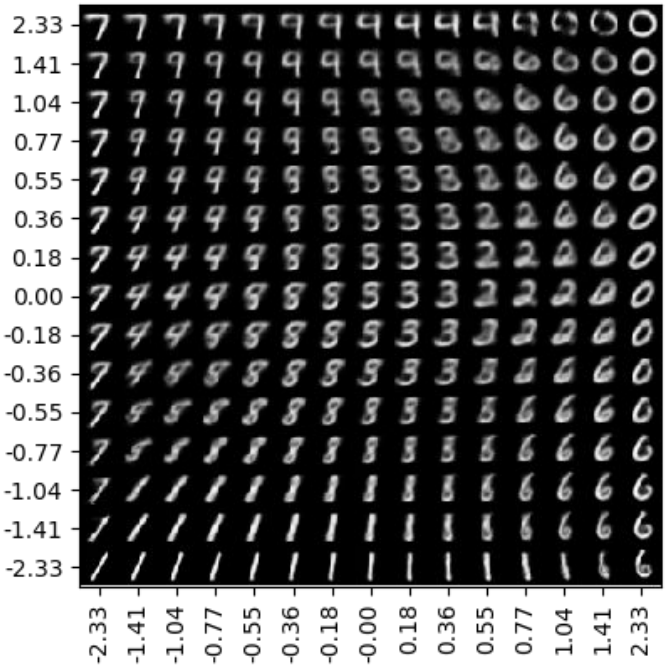
VAE - Reparamétrisation de la couche d'échantillonnage

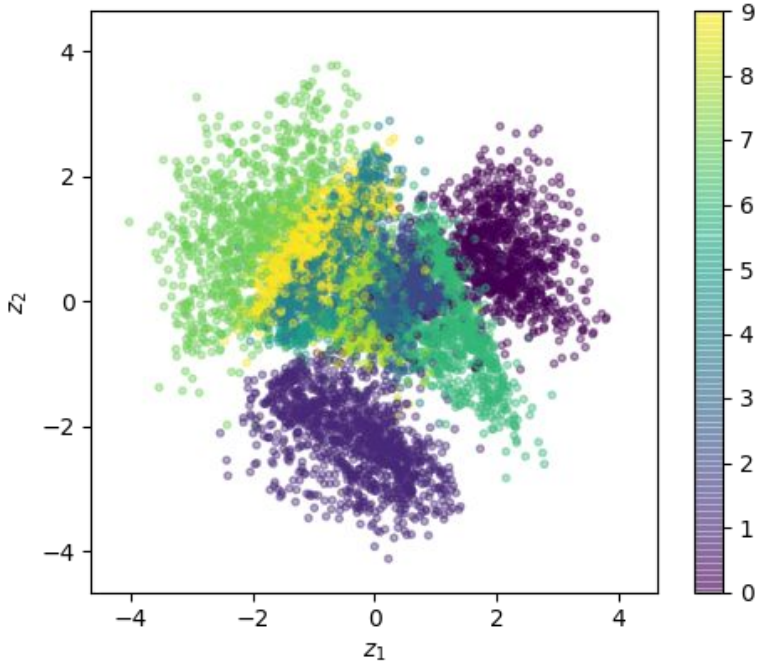


Backpropagation impossible

Backpropagation possible

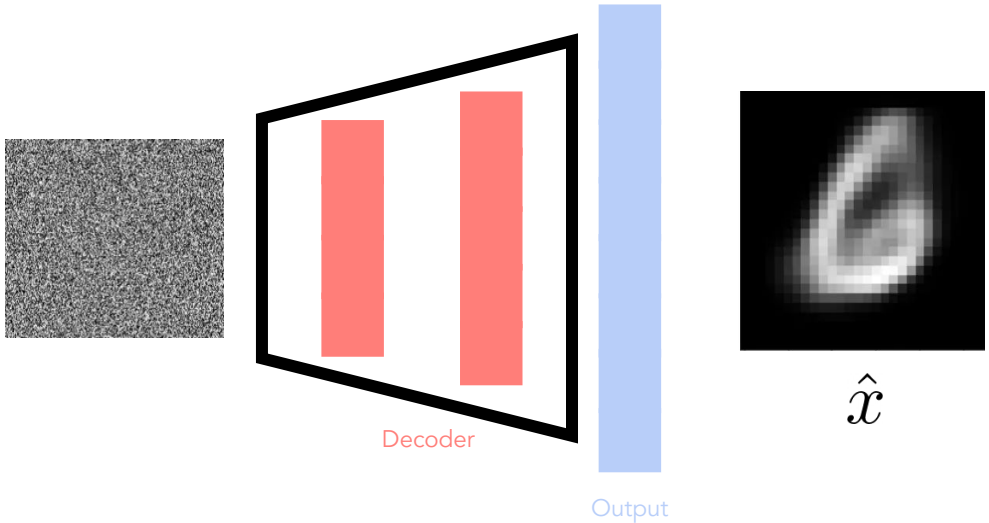
VAE - Exploration de l’espace latent



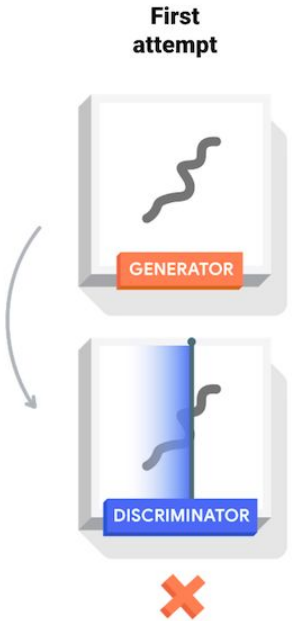


Generative Adversarial Networks (GANs)

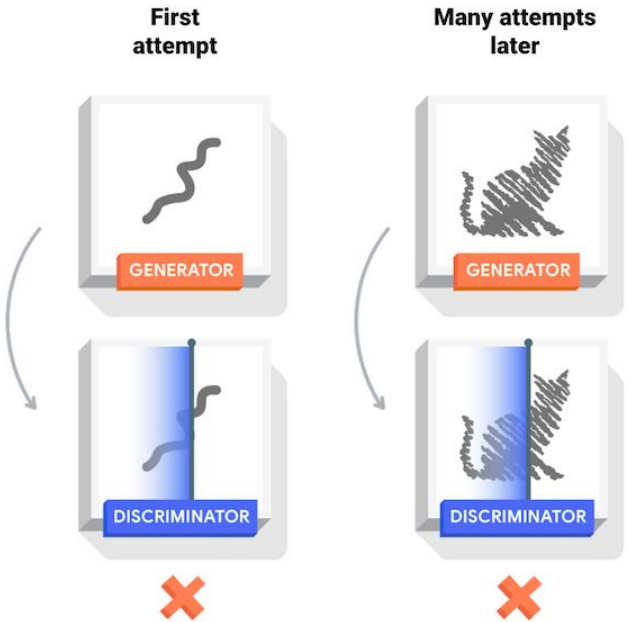
Generative Adversarial Networks (GANs)



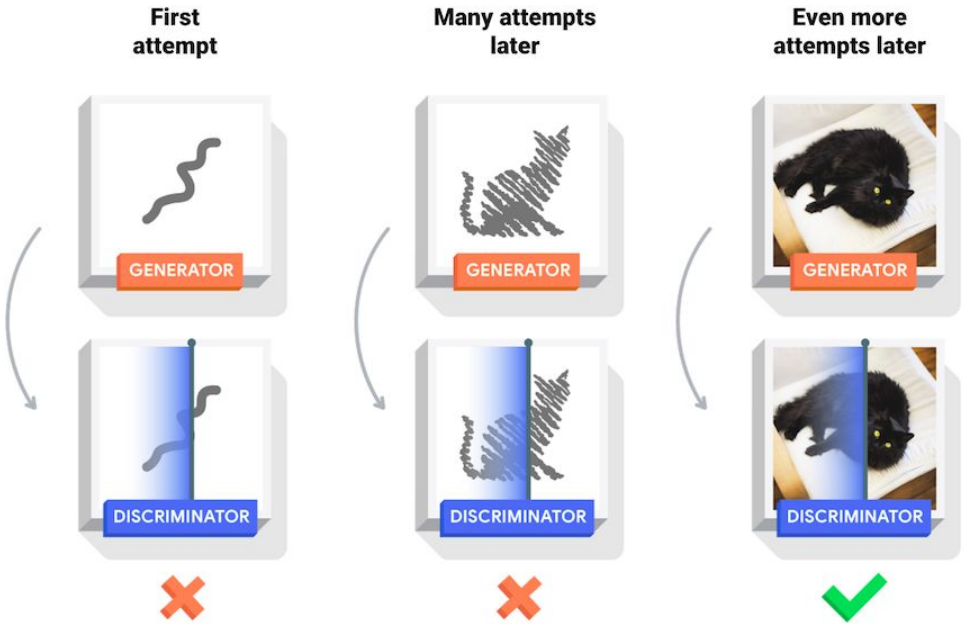
Generative Adversarial Networks (GANs)



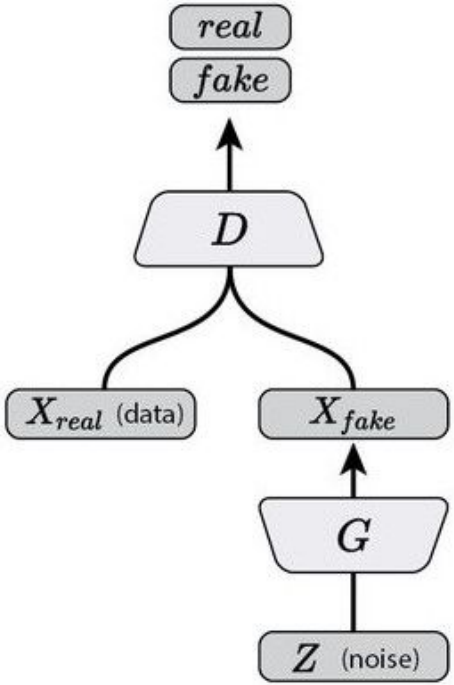
Generative Adversarial Networks (GANs)



Generative Adversarial Networks (GANs)



Generative Adversarial Networks (GANs)



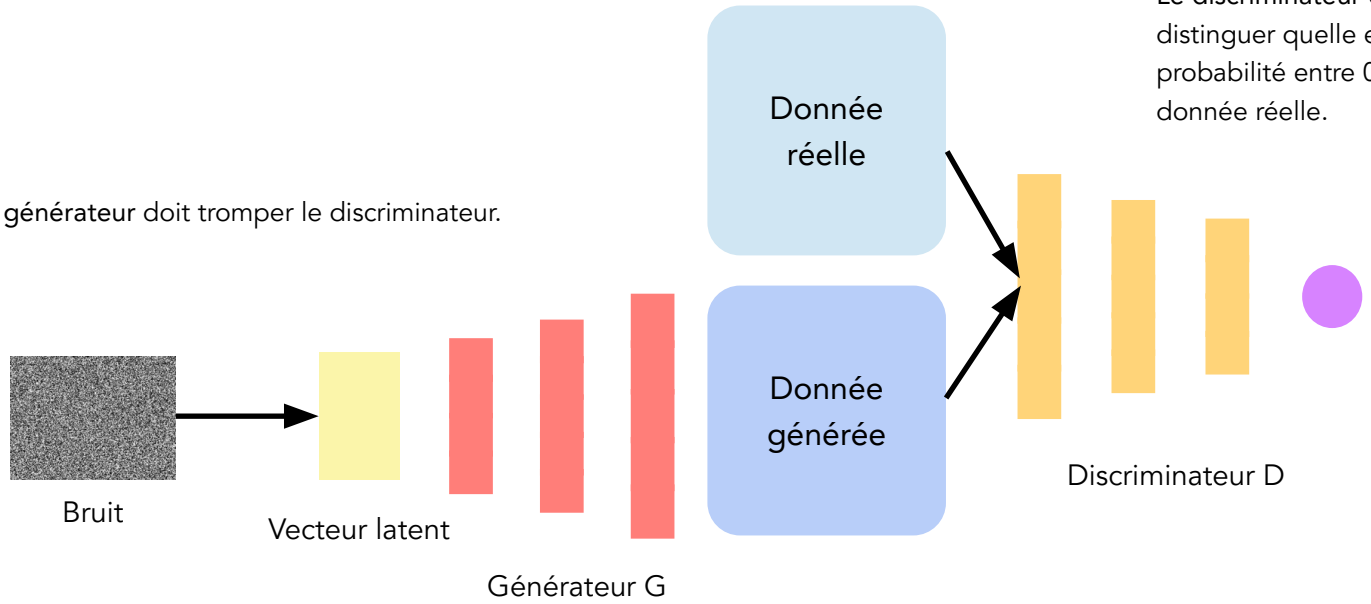
Source : Goodfellow, Ian, et al. Generative adversarial nets. NIPS 2014

GANs

Deux éléments principaux : un générateur et un discriminateur, qui sont formés de manière concurrentielle.

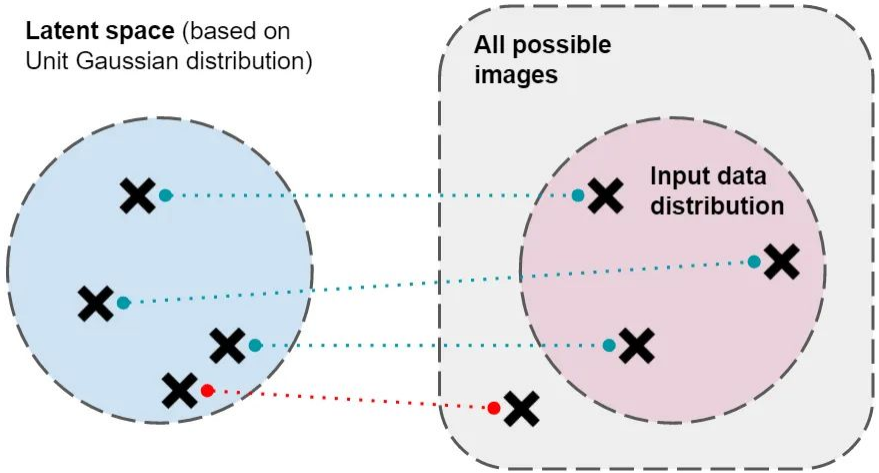
Le discriminateur est un classifieur binaire qui doit distinguer quelle est la donnée réelle. Il donne une probabilité entre 0 et 1 pour que l'élément soit une donnée réelle.

Le générateur doit tromper le discriminateur.

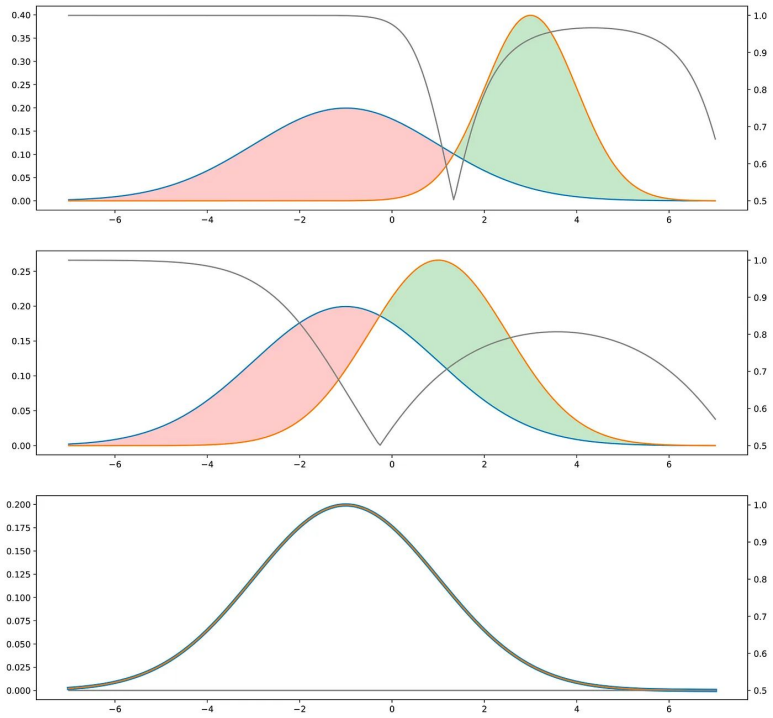


Intuition Générateur

Dans l'espace de toutes les images possibles, il existe un sous-espace plus petit associées aux images du jeu de données d'apprentissage. Le générateur sera pénalisé par le discriminateur pour avoir créé des images qui n'appartiennent pas à la distribution des données d'apprentissage.

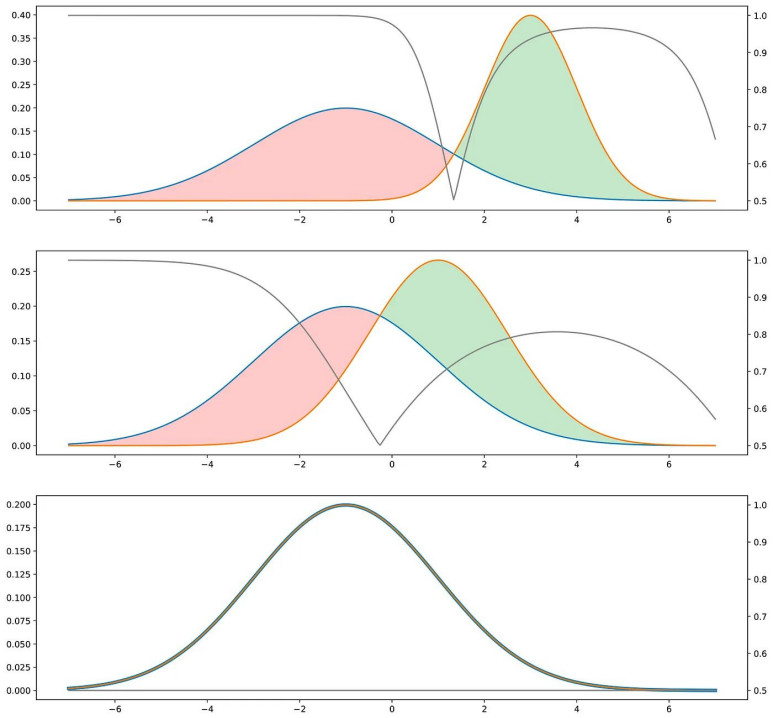


Intuition GANs



- La distribution bleue est associée aux images du jeu de données.
- La distribution orange est associée aux images générées.
- En gris, avec l'axe des y correspondant à droite, il est affiché la probabilité que le discriminateur soit vrai s'il choisit la classe ayant la densité la plus élevée dans chaque point (en supposant que les données "vraies" et "générées" sont en proportions égales).

Intuition GANs



- La distribution bleue est associée aux images du jeu de données.
- La distribution orange est associée aux images générées.
- En gris, avec l'axe des y correspondant à droite, il est affiché la probabilité que le discriminateur soit vrai s'il choisit la classe ayant la densité la plus élevée dans chaque point (en supposant que les données "vraies" et "générées" sont en proportions égales).

→ **“Transport optimal”** de distribution de probabilité
= Wasserstein GAN

GANs - Optimisation

Entraînement conjoint des deux réseaux G et D

“Compétition” Minimax dans un jeu à deux joueurs (chaque joueur est un réseau). Approche inspirée de la théorie des jeux.

GANs - Optimisation

Entraînement conjoint des deux réseaux G et D

"Compétition" Minimax dans un jeu à deux joueurs (chaque joueur est un réseau). Approche inspirée de la théorie des jeux.

$$\max_D L(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log(D(x))] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]$$

Maximise pour D(x) proche de 1

Maximise pour D(G(z)) proche de 0

GANs - Optimisation

Entraînement conjoint des deux réseaux G et D

"Compétition" Minimax dans un jeu à deux joueurs (chaque joueur est un réseau). Approche inspirée de la théorie des jeux.

$$\min_G \max_D L(D,G) = \mathbb{E}_{x \sim p_{data}(x)}[\log(D(x))] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

Minimise pour D(G(z)) proche de 1

GANs - Optimisation

Entraînement conjoint des deux réseaux G et D

"Compétition" Minimax dans un jeu à deux joueurs (chaque joueur est un réseau). Approche inspirée de la théorie des jeux.

$$\min_G \max_D L(D,G) = \mathbb{E}_{x \sim p_{data}(x)}[\log(D(x))] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

→ Montée de gradient pour le discriminateur

→ Descente de gradient pour le générateur

GANs - Optimisation

Entraînement conjoint des deux réseaux G et D

"Compétition" Minimax dans un jeu à deux joueurs (chaque joueur est un réseau). Approche inspirée de la théorie des jeux.

$$\min_G \max_D L(D,G) = \mathbb{E}_{x \sim p_{data}(x)}[\log(D(x))] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

→ Montée de gradient pour le discriminateur

→ Descente de gradient pour le générateur

GANs - Training

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

GANs - limitations

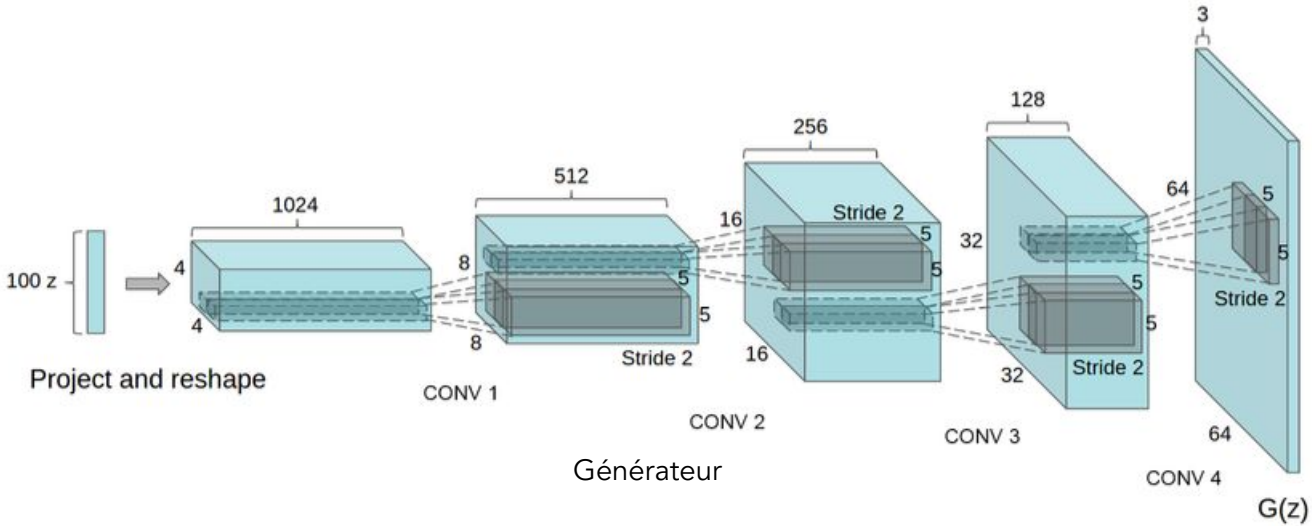
- Les GANs peuvent souffrir de mode collapse, lorsque le générateur produit une variété limitée d'échantillons, ignorant certains modes dans la distribution des données. Il peut en résulter un manque de diversité dans les données générées.

- Instabilité de l’entraînement : très difficile à paramétrer, peut souffrir de vanishing gradient.

- L’évaluation des GANs peut-être compliquée.

- Les GANs nécessitent souvent un temps d'apprentissage prolongé, en particulier pour la génération d'images à haute résolution. Cette longue durée d'apprentissage peut rendre l'expérimentation et l'itération fastidieuses.

Convolutional GAN (CGAN)

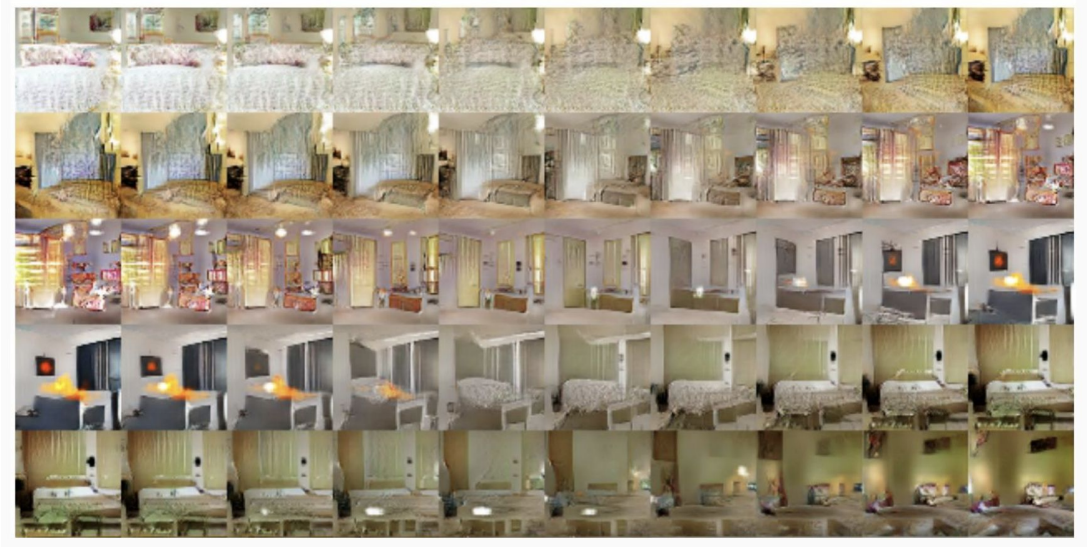


Générateur

Résultats Génération

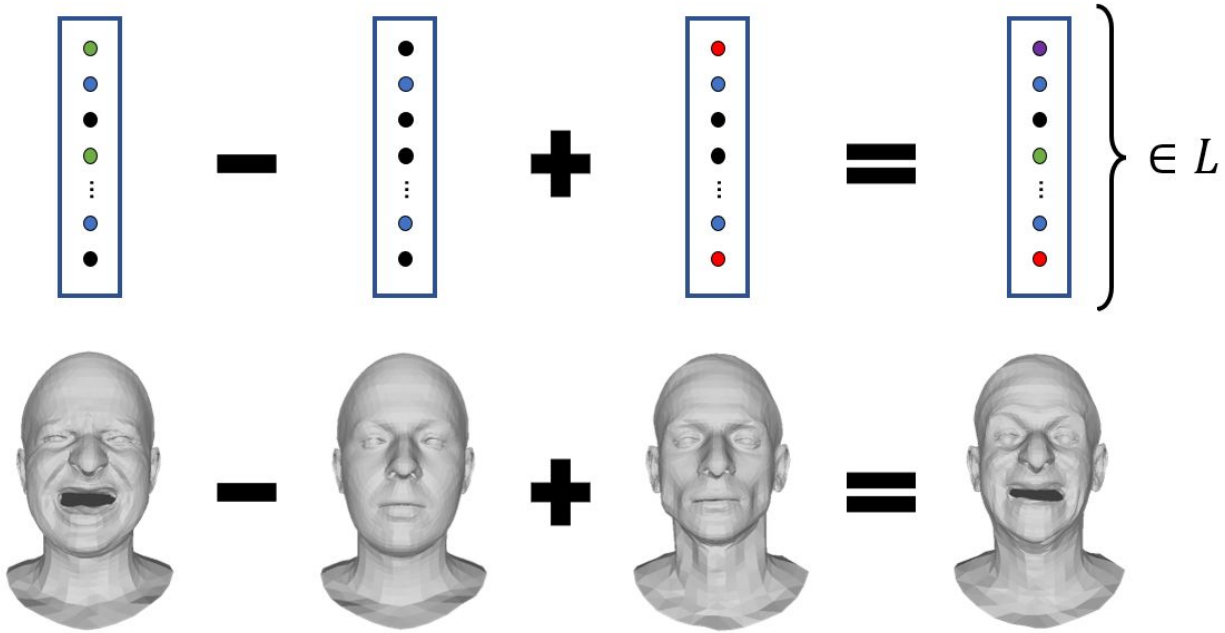
Entraîné sur le jeu de données LSUN

Interpolation sur l’espace latent

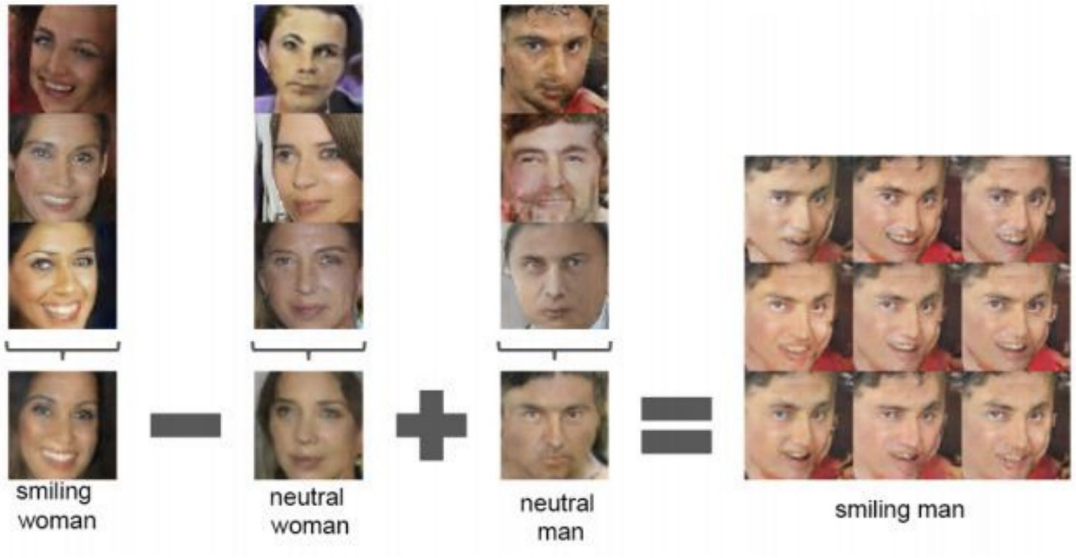


Transition assez smooth entre les images générées

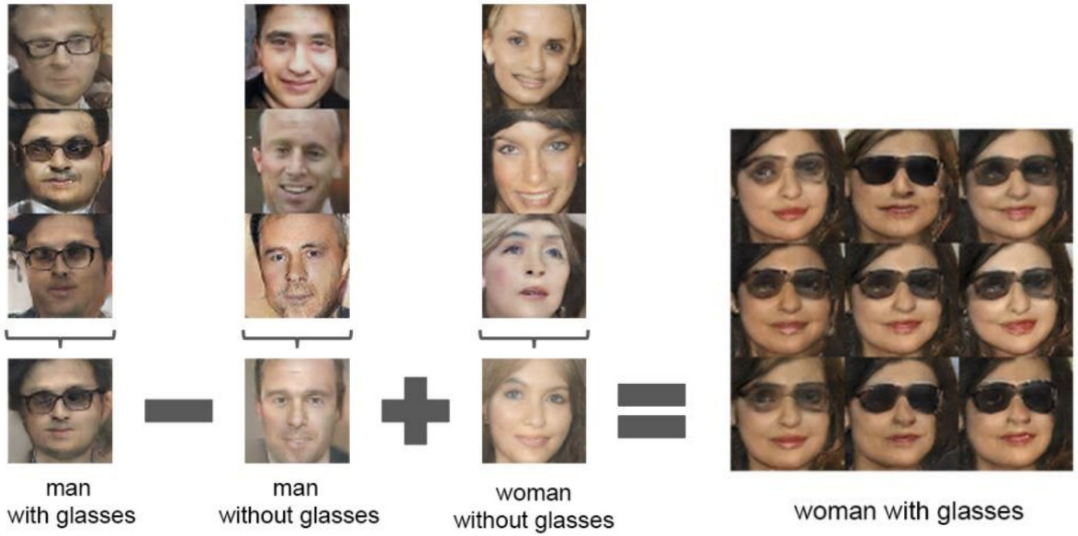
Opération sur l’espace latent



Opération sur l’espace latent

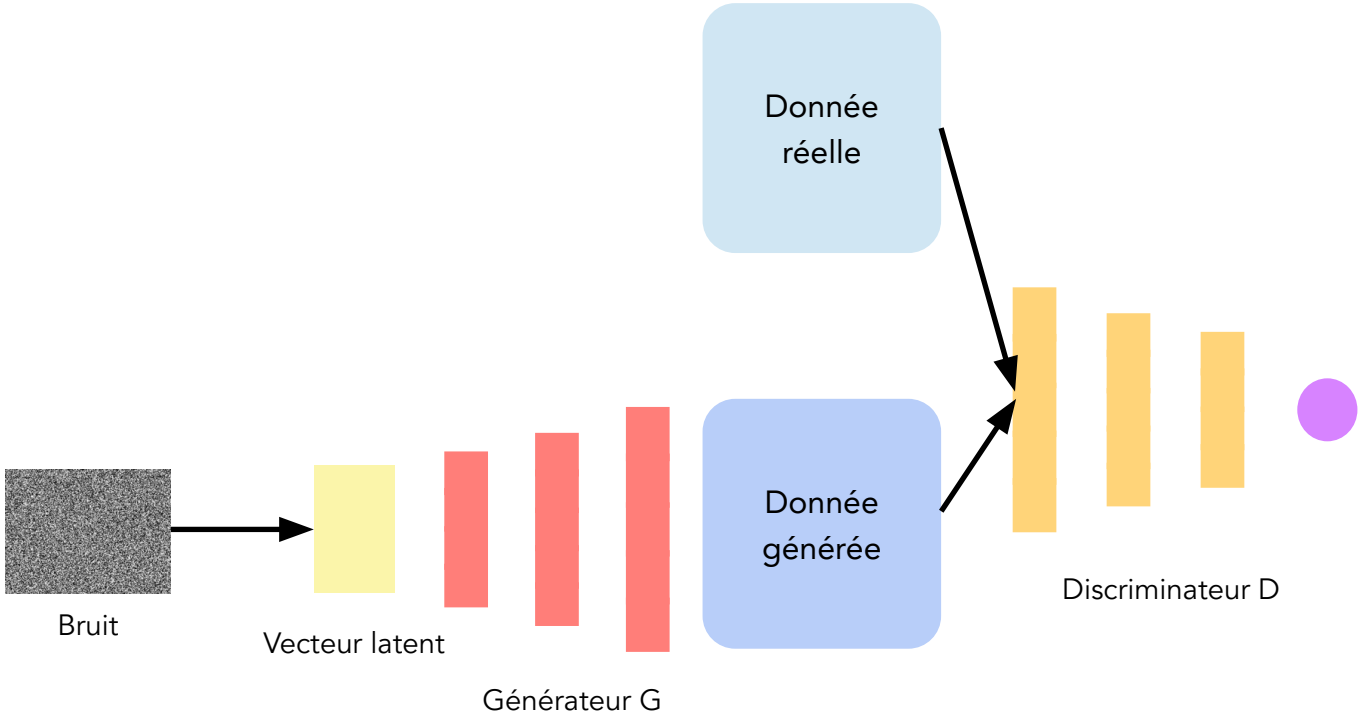


Opération sur l’espace latent

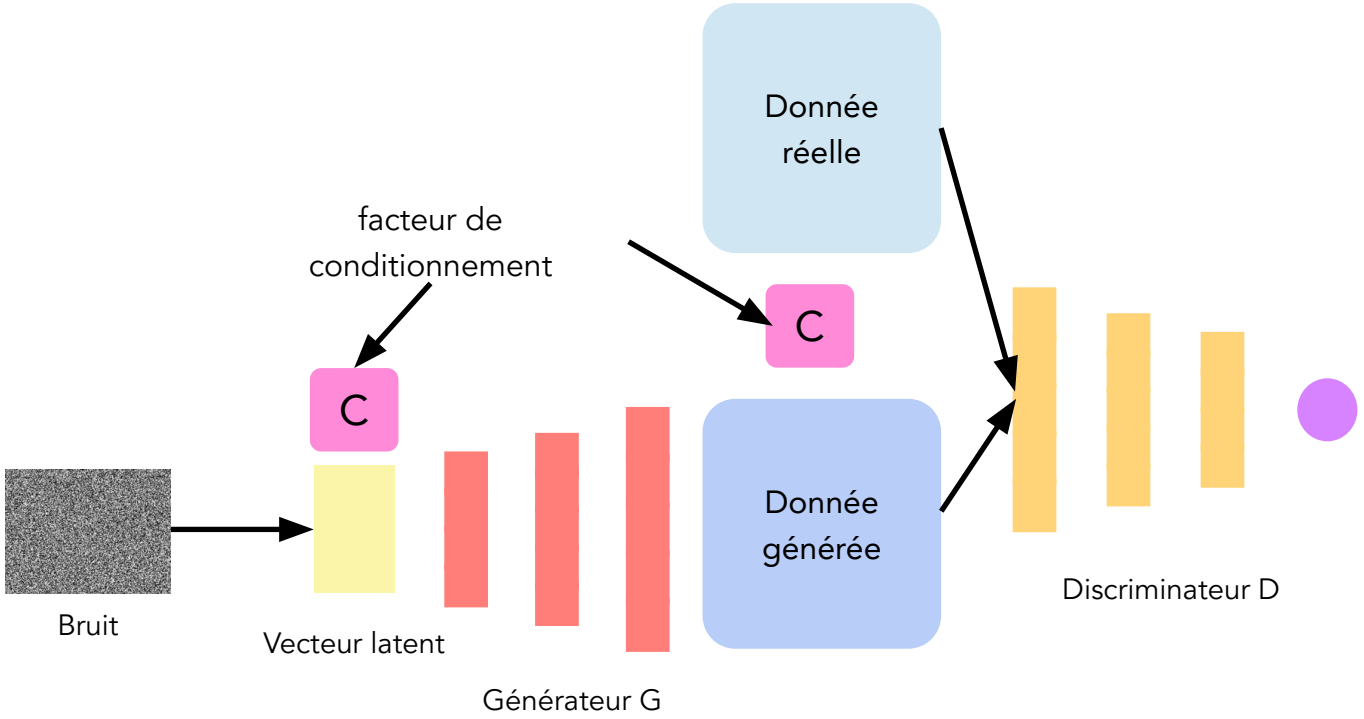


Opération sur l’espace latent

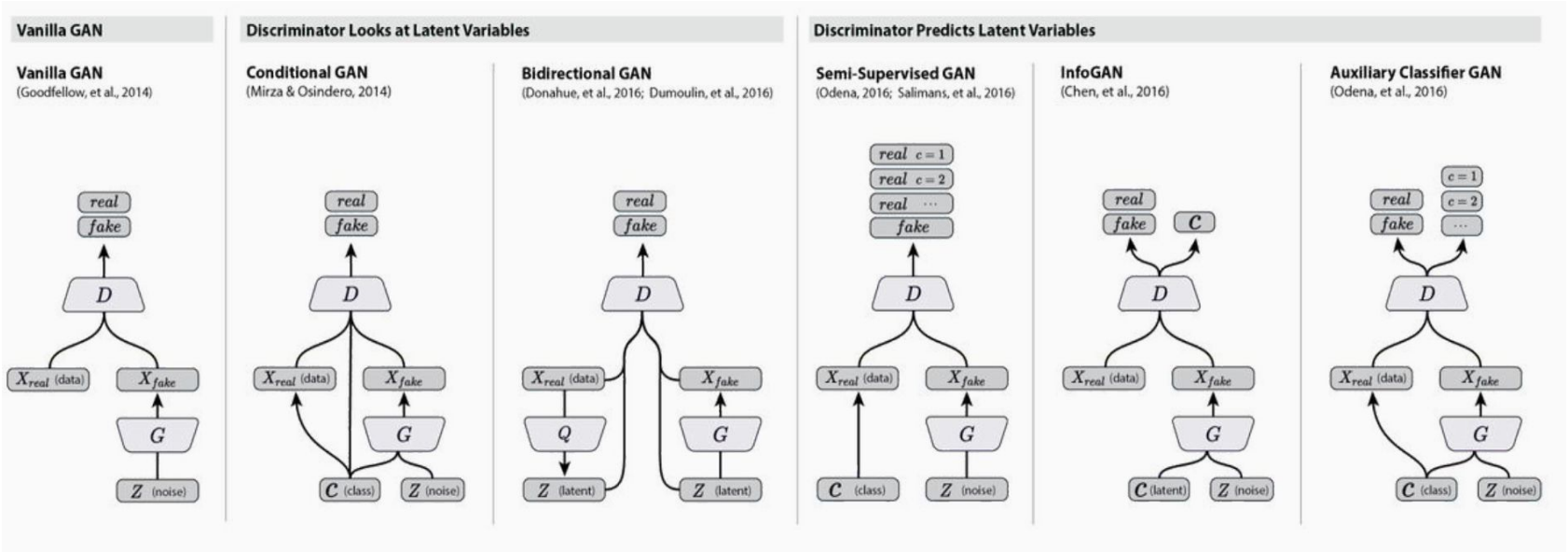
Construire d**ifférent types de GAN**



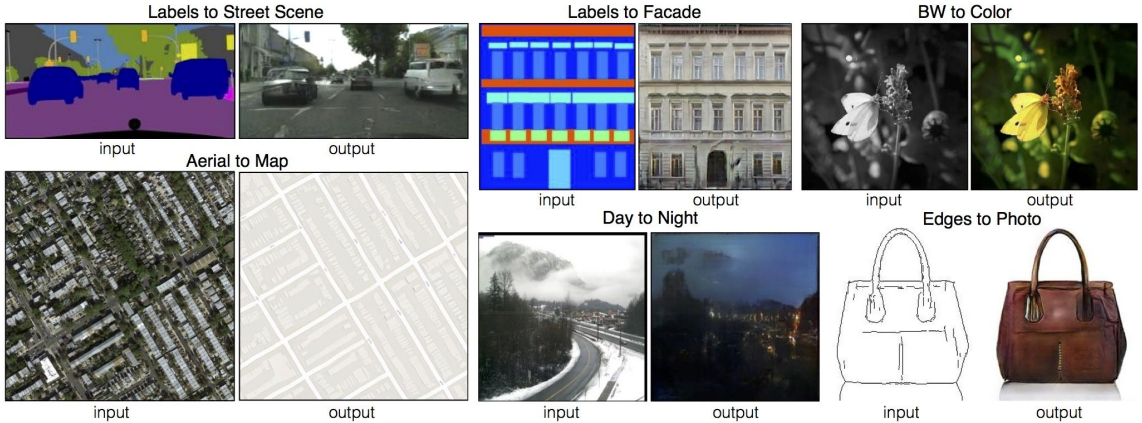
Construire d**ifférent types de GAN**



Différent types de GAN



Pix2pix



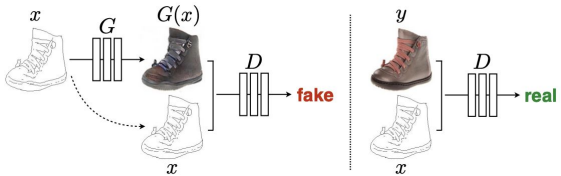
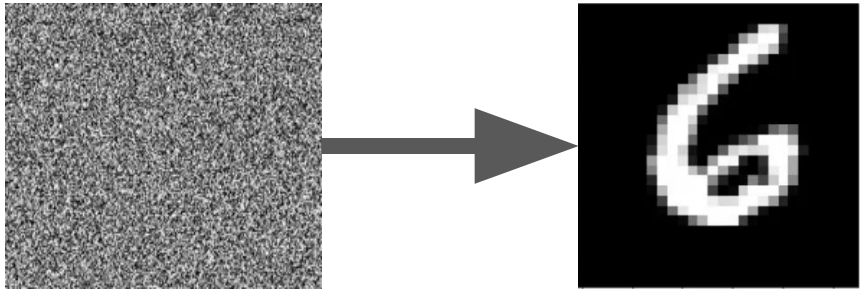


Figure 2: Training a conditional GAN to map edges→photo. The discriminator, D, learns to classify between fake (synthesized by the generator) and real {edge, photo} tuples. The generator, G, learns to fool the discriminator. Unlike an unconditional GAN, both the generator and discriminator observe the input edge map.

CycleGAN

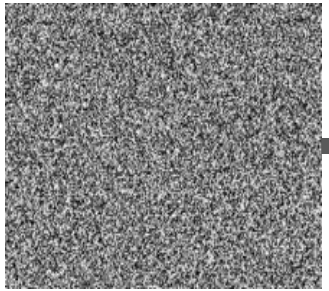
Noise space *Data space*



GAN classique

CycleGAN

Noise space



Data space

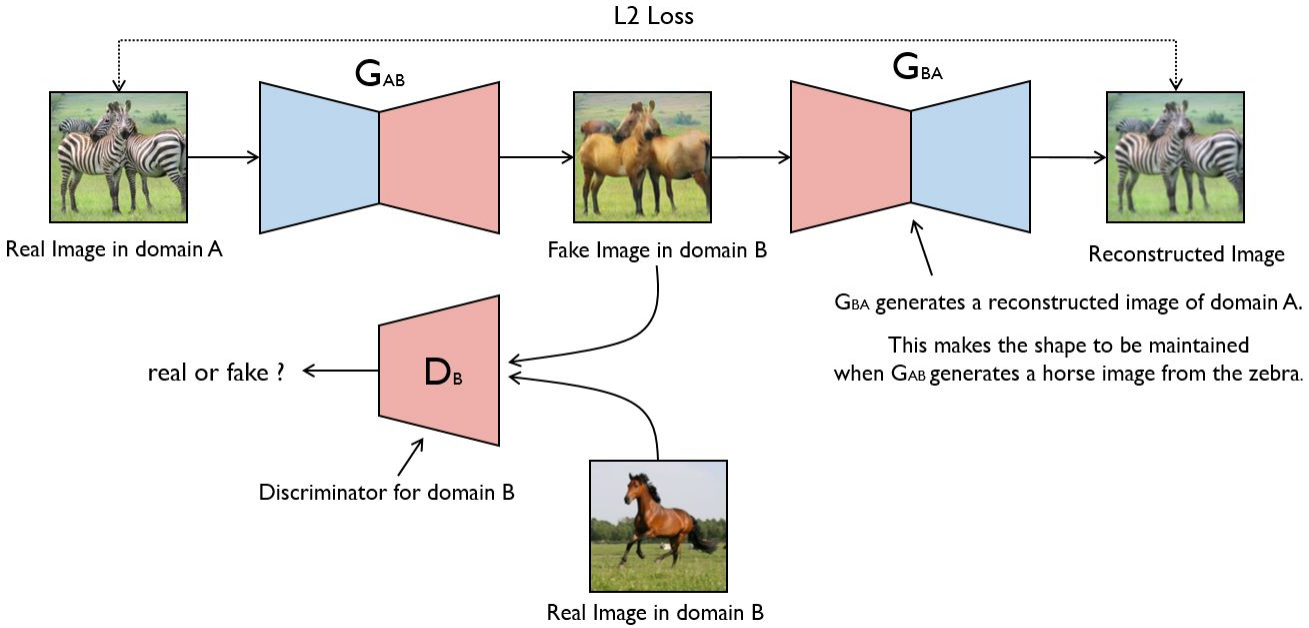
GAN classique

Data space A

Data space B

CycleGAN

CycleGAN


L2 Loss
G_{AB}
G_{BA}
Real Image in domain A
Fake Image in domain B
Reconstructed Image
G_{BA} generates a reconstructed image of domain A.
This makes the shape to be maintained
when G_{AB} generates a horse image from the zebra.
real or fake ?
D_B
Discriminator for domain B
Real Image in domain B

CycleGAN

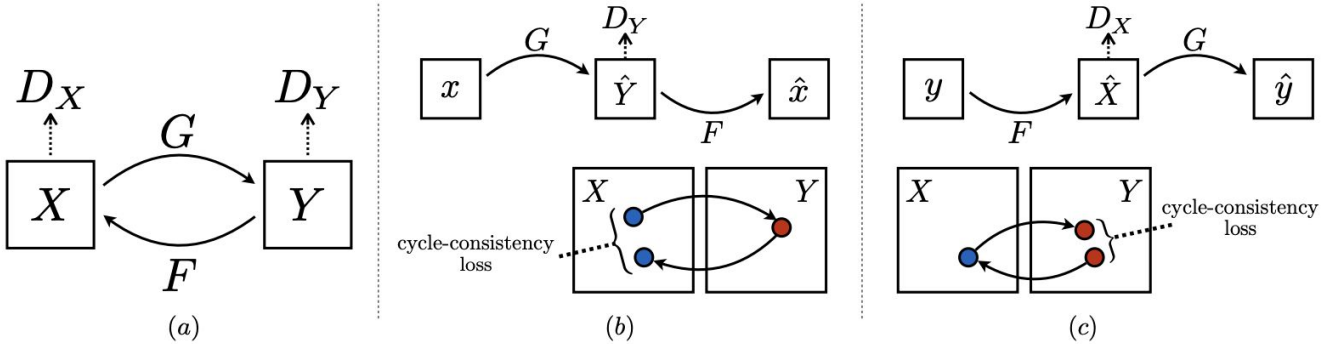
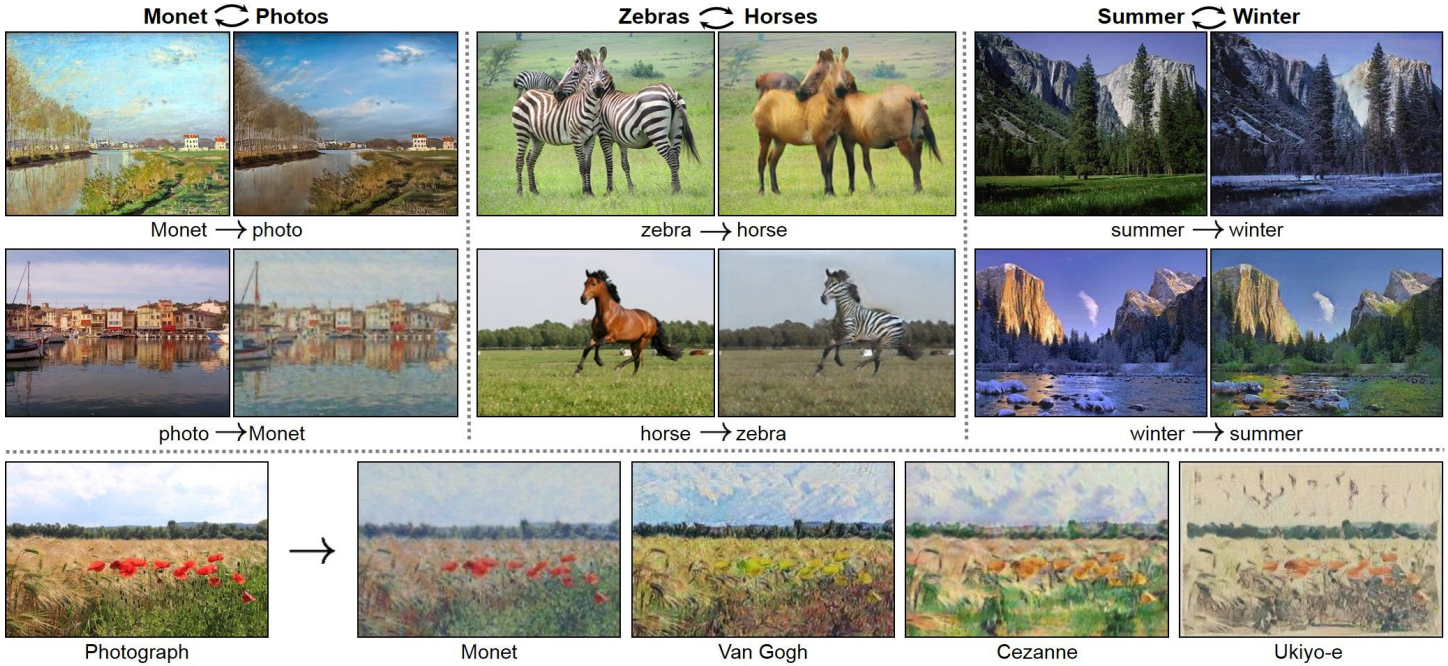


Figure 3: (a) Our model contains two mapping functions $G : X \rightarrow Y$ and $F : Y \rightarrow X$, and associated adversarial discriminators D_Y and D_X. D_Y encourages G to translate X into outputs indistinguishable from domain Y, and vice versa for D_X and F. To further regularize the mappings, we introduce two *cycle consistency losses* that capture the intuition that if we translate from one domain to the other and back again we should arrive at where we started: (b) forward cycle-consistency loss: $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$, and (c) backward cycle-consistency loss: $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$

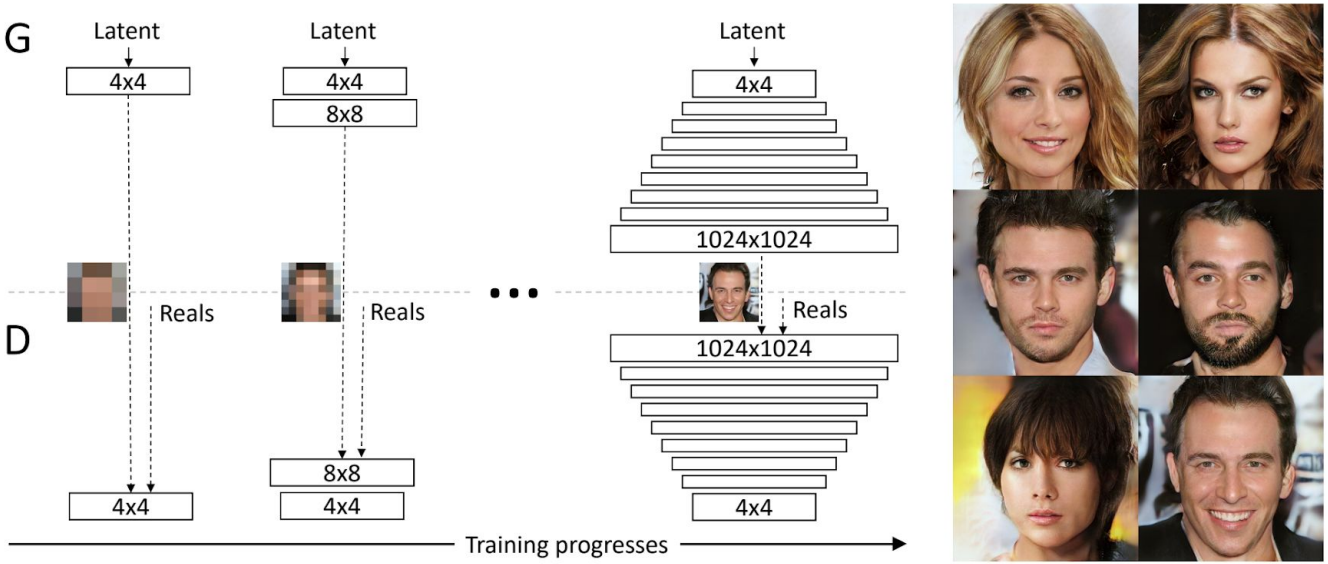
CycleGAN


Monet ⇄ Photos
Zebras ⇄ Horses
Summer ⇄ Winter
Monet → photo
zebra → horse
summer → winter
photo → Monet
horse → zebra
winter → summer
Photograph
Monet
Van Gogh
Cezanne
Ukiyo-e

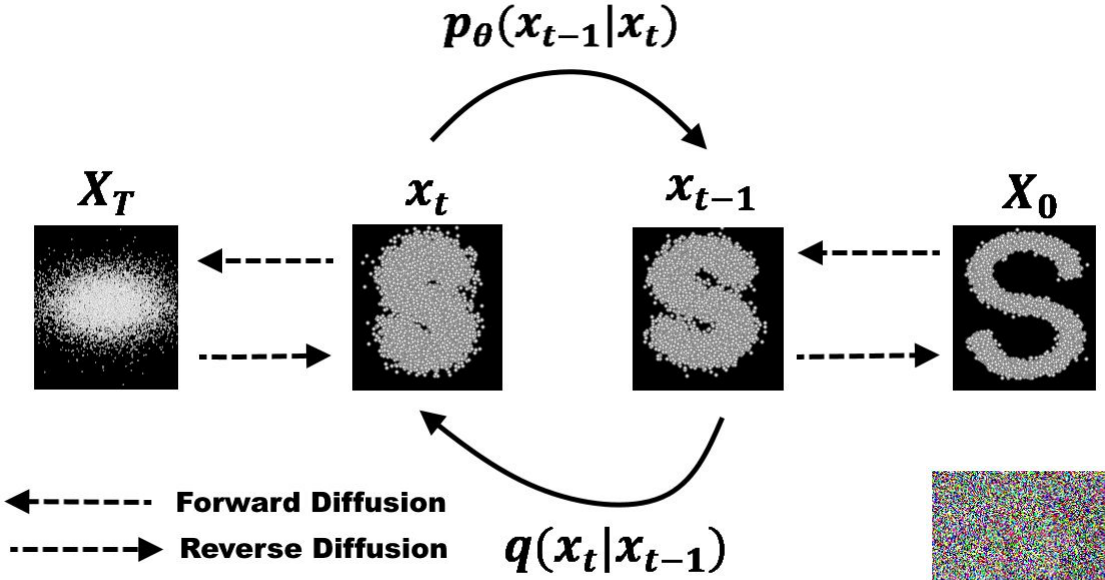
Progressive Growing of GAN (PGGAN)

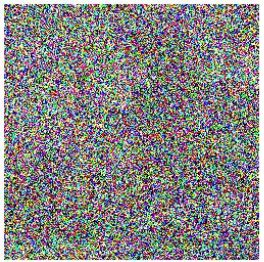
- Croissance progressive : La croissance progressive consiste à entraîner un GAN étape par étape, en partant d'une faible résolution et en augmentant progressivement la résolution.
- Génération d’images 1024x1024

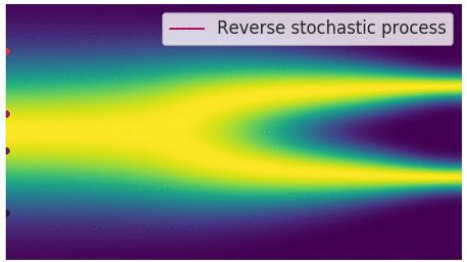
PGGAN


G
Latent
4x4
Latent
4x4
8x8
Latent
4x4
1024x1024
D
Reals
Reals
Reals
1024x1024
8x8
4x4
4x4
4x4
Training progresses

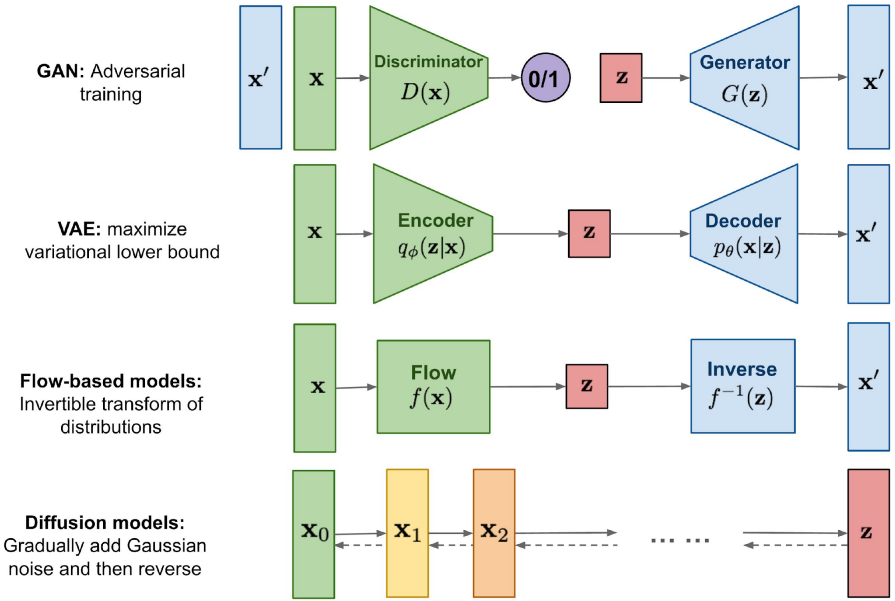
Modèle de diffusion







Vue d'ensemble des principaux modèles génératifs



Source :
https://lilianweng.github.io/posts/2021-07-11-diffusion-models/